

# Document made available under the Patent Cooperation Treaty (PCT)

International application number: PCT/JP2005/015815

International filing date: 30 August 2005 (30.08.2005)

Document type: Certified copy of priority document

Document details: Country/Office: JP  
Number: 2004-256465  
Filing date: 03 September 2004 (03.09.2004)

Date of receipt at the International Bureau: 13 October 2005 (13.10.2005)

Remark: Priority document submitted or transmitted to the International Bureau in compliance with Rule 17.1(a) or (b)



World Intellectual Property Organization (WIPO) - Geneva, Switzerland  
Organisation Mondiale de la Propriété Intellectuelle (OMPI) - Genève, Suisse

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日  
Date of Application: 2 0 0 4 年 9 月 3 日

出 願 番 号  
Application Number: 特 願 2 0 0 4 - 2 5 6 4 6 5

パリ条約による外国への出願  
に用いる優先権の主張の基礎  
となる出願の国コードと出願  
番号  
J P 2 0 0 4 - 2 5 6 4 6 5  
The country code and number  
of your priority application,  
to be used for filing abroad  
under the Paris Convention, is

出 願 人  
Applicant(s): ソニー株式会社

2 0 0 5 年 9 月 2 8 日

特許庁長官  
Commissioner,  
Japan Patent Office

中 嶋



【書類名】	特許願
【整理番号】	0490463303
【提出日】	平成16年 9月 3日
【あて先】	特許庁長官殿
【国際特許分類】	G09C 1/00
【発明者】	
【住所又は居所】	東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社内
【氏名】	白井 太三
【発明者】	
【住所又は居所】	ベルギー国、ケイ・ユー・ルーヴァン リサーチアンドデベロップメント フロート ベゲインホフ 5 8－5 9 B－3 0 0 0 ルーヴァン ルーベンカソリック大学
【氏名】	バート プレネール
【特許出願人】	
【識別番号】	000002185
【氏名又は名称】	ソニー株式会社
【代理人】	
【識別番号】	100093241
【弁理士】	
【氏名又は名称】	宮田 正昭
【電話番号】	03-5541-7577
【選任した代理人】	
【識別番号】	100101801
【弁理士】	
【氏名又は名称】	山田 英治
【電話番号】	03-5541-7577
【選任した代理人】	
【識別番号】	100086531
【弁理士】	
【氏名又は名称】	澤田 俊夫
【電話番号】	03-5541-7577
【手数料の表示】	
【予納台帳番号】	048747
【納付金額】	16,000円
【提出物件の目録】	
【物件名】	特許請求の範囲 1
【物件名】	明細書 1
【物件名】	図面 1
【物件名】	要約書 1
【包括委任状番号】	9904833

## 【書類名】 特許請求の範囲

### 【請求項 1】

F e i s t e l 型共通鍵ブロック暗号処理を実行する暗号処理装置であり、

非線形変換部および線形変換部を有する S P N 型の F 関数を、複数ラウンド繰り返し実行する構成を有し、

前記複数ラウンド各々に対応する F 関数の線形変換部は、m 個の非線形変換部各々の出力する n ビット、総計 m n ビットの入力に対する線形変換処理を、正方 M D S ( M a x i m u m D i s t a n c e S e p a r a b l e ) 行列を適用した線形変換処理として実行する構成であり、

少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々においては、異なる正方 M D S 行列：L a , L b が適用され、かつ該正方 M D S 行列の逆行列：L a <sup>-1</sup> , L b <sup>-1</sup> を構成する列ベクトルから任意に選択した m 個の列ベクトルによって構成する行列が線形独立であることを特徴とする暗号処理装置。

### 【請求項 2】

前記暗号処理装置において、さらに、

前記逆行列：L a <sup>-1</sup> , L b <sup>-1</sup> を構成する列ベクトルから任意に選択した m 個の列ベクトルによって構成する行列が正方 M D S 行列であることを特徴とする請求項 1 に記載の暗号処理装置。

### 【請求項 3】

前記 F e i s t e l 型共通鍵ブロック暗号処理のアルゴリズムは、ラウンド数 2 r の暗号処理アルゴリズムであり、

前記 F 関数の線形変換部は、

r 個の全ての偶数ラウンドおよび r 個の全ての奇数ラウンドにおいて  $2 \leq q < r$  の q 種類の異なる正方 M D S 行列を順次繰り返し適用した線形変換処理を実行する構成であることを特徴とする請求項 1 に記載の暗号処理装置。

### 【請求項 4】

前記 F 関数の線形変換部において適用する異なる複数の正方 M D S 行列の各々は、該複数の正方 M D S 行列を構成する列ベクトルから任意に選択した m 個の列ベクトルによって構成する行列が線形独立である正方 M D S 行列であることを特徴とする請求項 1 に記載の暗号処理装置。

### 【請求項 5】

前記 F 関数の線形変換部において適用する異なる複数の正方 M D S 行列の各々は、該複数の正方 M D S 行列を構成する列ベクトルから任意に選択した m 個の列ベクトルによって構成する行列も正方 M D S 行列となる正方 M D S 行列であることを特徴とする請求項 1 に記載の暗号処理装置。

### 【請求項 6】

前記 F 関数の線形変換部において適用する異なる複数の正方 M D S 行列の各々は、該複数の正方 M D S 行列を構成する要素を全て含む正方 M D S 行列 M から選択された行ベクトルによって構成される行列 M' から抽出された列ベクトルによって構成される行列によって構成されていることを特徴とする請求項 1 に記載の暗号処理装置。

### 【請求項 7】

F e i s t e l 型共通鍵ブロック暗号処理を実行する暗号処理方法であり、

非線形変換処理および線形変換処理を実行する S P N 型の F 関数を、複数ラウンド繰り返し実行し、

前記複数ラウンド各々に対応する F 関数の線形変換処理は、m 個の非線形変換部各々の出力する n ビット、総計 m n ビットの入力に対する線形変換処理を、正方 M D S ( M a x i m u m D i s t a n c e S e p a r a b l e ) 行列を適用した線形変換処理として実行し、

少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々においては、異なる正方 M D S 行列：L a , L b が適用され、かつ該正方 M D S 行列の逆行列：L a <sup>-1</sup> ,

$Lb^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立である正方MDS行列による線形変換処理を実行することを特徴とする暗号処理方法。

【請求項 8】

前記暗号処理方法において、さらに、

前記逆行列： $La^{-1}$ ， $Lb^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が正方MDS行列である正方MDS行列による線形変換処理を実行することを特徴とする請求項 7 に記載の暗号処理方法。

【請求項 9】

前記Feistel型共通鍵ブロック暗号処理のアルゴリズムは、ラウンド数 $2r$ の暗号処理アルゴリズムであり、

前記F関数の線形変換処理は、

$r$ 個の全ての偶数ラウンドおよび $r$ 個の全ての奇数ラウンドにおいて $2 \leq q < r$ の $q$ 種類の異なる正方MDS行列を順次繰り返し適用した線形変換処理を実行することを特徴とする請求項 7 に記載の暗号処理方法。

【請求項 10】

前記F関数の線形変換処理において適用する異なる複数の正方MDS行列の各々は、該複数の正方MDS行列を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立である正方MDS行列であることを特徴とする請求項 7 に記載の暗号処理方法。

【請求項 11】

前記F関数の線形変換処理において適用する異なる複数の正方MDS行列の各々は、該複数の正方MDS行列を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列も正方MDS行列となる正方MDS行列であることを特徴とする請求項 7 に記載の暗号処理方法。

【請求項 12】

前記F関数の線形変換処理において適用する異なる複数の正方MDS行列の各々は、該複数の正方MDS行列を構成する要素を全て含む正方MDS行列 $M$ から選択された行ベクトルによって構成される行列 $M'$ から抽出された列ベクトルによって構成される行列によって構成されていることを特徴とする請求項 7 に記載の暗号処理方法。

【請求項 13】

Feistel型共通鍵ブロック暗号処理を実行するコンピュータ・プログラムであり、

非線形変換処理および線形変換処理を実行するSPN型のF関数を、複数ラウンド繰り返し実行するステップを有し、

前記複数ラウンド各々に対応するF関数の線形変換処理は、 $m$ 個の非線形変換部各々の出力する $n$ ビット、総計 $mn$ ビットの入力に対する線形変換処理を、正方MDS (Maximum Distance Separable) 行列を適用した線形変換処理として実行する線形変換ステップであり、

前記線形変換ステップにおいて、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々では、異なる正方MDS行列： $La$ ， $Lb$ が適用され、かつ該正方MDS行列の逆行列： $La^{-1}$ ， $Lb^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立である正方MDS行列による線形変換処理を実行することを特徴とするコンピュータ・プログラム。

【書類名】 明細書

【発明の名称】 暗号処理装置、および暗号処理方法、並びにコンピュータ・プログラム

【技術分野】

【0001】

本発明は、暗号処理装置、および暗号処理方法、並びにコンピュータ・プログラムに関する。さらに詳細には、暗号解析処理、攻撃処理として知られる線形解析、差分解析に対する耐性を向上させた暗号処理装置、および暗号処理方法、並びにコンピュータ・プログラムに関する。

【背景技術】

【0002】

昨今、ネットワーク通信、電子商取引の発展に伴い、通信におけるセキュリティ確保が重要な問題となっている。セキュリティ確保の1つの方法が暗号技術であり、現在、様々な暗号化手法を用いた通信が実際に行なわれている。

【0003】

例えばICカード等の小型の装置中に暗号処理モジュールを埋め込み、ICカードと、データ読み取り書き込み装置としてのリーダライタとの間でデータ送受信を行ない、認証処理、あるいは送受信データの暗号化、復号を行なうシステムが実用化されている。

【0004】

暗号処理アルゴリズムには様々なものがあるが、大きく分類すると、暗号化鍵と復号鍵を異なる鍵、例えば公開鍵と秘密鍵として設定する公開鍵暗号方式と、暗号化鍵と復号鍵を共通の鍵として設定する共通鍵暗号方式とに分類される。

【0005】

共通鍵暗号方式にも様々なアルゴリズムがあるが、その1つに共通鍵をベースとして複数の鍵を生成して、生成した複数の鍵を用いてブロック単位（64ビット、128ビットなど）のデータ変換処理を繰り返し実行する方式がある。このような鍵生成方式とデータ変換処理を適用したアルゴリズムの代表的なものが共通鍵ブロック暗号方式である。

【0006】

代表的な共通鍵ブロック暗号のアルゴリズムとしては、例えば米国標準暗号としてのDES（Data Encryption Standard）アルゴリズムがあり、様々な分野において広く用いられている。

【0007】

DESに代表される共通鍵ブロック暗号のアルゴリズムは、主として、入力データの変換を実行するラウンド関数部と、ラウンド関数（F関数）部の各ラウンドで適用する鍵を生成する鍵スケジュール部とに分けることができる。ラウンド関数部の各ラウンドで適用するラウンド鍵（副鍵）は、1つのマスター鍵（主鍵）に基づいて、鍵スケジュール部に入力されて生成され、各ラウンド関数部で適用される。

【0008】

しかし、このような共通鍵暗号処理においては、暗号解析による鍵の漏洩が問題となっている。暗号解析または攻撃手法の代表的な手法として、ある差分を持つ入力データ（平文）とその出力データ（暗号文）を多数解析することにより各ラウンド関数における適用鍵を解析する差分解析（差分解読法または差分攻撃とも呼ばれる）や、平文と対応暗号文に基づく解析を行う線形解析（線形解読法または線形攻撃とも呼ばれる）が知られている。

【0009】

暗号解析による鍵の解析が容易であるということは、その暗号処理の安全性が低いということになる。従来のDESアルゴリズムにおいては、ラウンド関数（F関数）部の線形変換部において適用する処理（変換行列）が、各段のラウンドにおいて等しいものであったため解析が行いやすく、結果として鍵の解析の容易性を招いているという問題がある。

【発明の開示】

【発明が解決しようとする課題】

#### 【0010】

本発明は、上記問題点に鑑みてなされたものであり、線形解析や差分解析に対する耐性の高い共通鍵ブロック暗号アルゴリズムを実現する暗号処理装置、および暗号処理方法、並びにコンピュータ・プログラムを提供することを目的とする。

#### 【課題を解決するための手段】

#### 【0011】

本発明の第1の側面は、

F e i s t e l 型共通鍵ブロック暗号処理を実行する暗号処理装置であり、

非線形変換部および線形変換部を有するS P N型のF関数を、複数ラウンド繰り返し実行する構成を有し、

前記複数ラウンド各々に対応するF関数の線形変換部は、m個の非線形変換部各々の出力するnビット、総計mnビットの入力に対する線形変換処理を、正方M D S (M a x i m u m D i s t a n c e S e p a r a b l e) 行列を適用した線形変換処理として実行する構成であり、

少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々においては、異なる正方M D S 行列：L a , L b が適用され、かつ該正方M D S 行列の逆行列：L a<sup>-1</sup>, L b<sup>-1</sup>を構成する列ベクトルから任意に選択したm個の列ベクトルによって構成する行列が線形独立であることを特徴とする暗号処理装置にある。

#### 【0012】

さらに、本発明の暗号処理装置の一実施態様において、さらに、前記逆行列：L a<sup>-1</sup>, L b<sup>-1</sup>を構成する列ベクトルから任意に選択したm個の列ベクトルによって構成する行列が正方M D S 行列であることを特徴とする。

#### 【0013】

さらに、本発明の暗号処理装置の一実施態様において、前記F e i s t e l 型共通鍵ブロック暗号処理のアルゴリズムは、ラウンド数2 rの暗号処理アルゴリズムであり、前記F関数の線形変換部は、r個の全ての偶数ラウンドおよびr個の全ての奇数ラウンドにおいて2 ≤ q < rのq種類の異なる正方M D S 行列を順次繰り返し適用した線形変換処理を実行する構成であることを特徴とする。

#### 【0014】

さらに、本発明の暗号処理装置の一実施態様において、前記F関数の線形変換部において適用する異なる複数の正方M D S 行列の各々は、該複数の正方M D S 行列を構成する列ベクトルから任意に選択したm個の列ベクトルによって構成する行列が線形独立である正方M D S 行列であることを特徴とする。

#### 【0015】

さらに、本発明の暗号処理装置の一実施態様において、前記F関数の線形変換部において適用する異なる複数の正方M D S 行列の各々は、該複数の正方M D S 行列を構成する列ベクトルから任意に選択したm個の列ベクトルによって構成する行列も正方M D S 行列となる正方M D S 行列であることを特徴とする。

#### 【0016】

さらに、本発明の暗号処理装置の一実施態様において、前記F関数の線形変換部において適用する異なる複数の正方M D S 行列の各々は、該複数の正方M D S 行列を構成する要素を全て含む正方M D S 行列Mから選択された行ベクトルによって構成される行列M'から抽出された列ベクトルによって構成される行列によって構成されていることを特徴とする。

#### 【0017】

さらに、本発明の第2の側面は、

F e i s t e l 型共通鍵ブロック暗号処理を実行する暗号処理方法であり、

非線形変換処理および線形変換処理を実行するS P N型のF関数を、複数ラウンド繰り返し実行し、

前記複数ラウンド各々に対応するF関数の線形変換処理は、m個の非線形変換部各々の

出力する $n$ ビット、総計 $mn$ ビットの入力に対する線形変換処理を、正方MDS (Maximum Distance Separable) 行列を適用した線形変換処理として実行し、

少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々においては、異なる正方MDS行列： $L_a$ 、 $L_b$ が適用され、かつ該正方MDS行列の逆行列： $L_a^{-1}$ 、 $L_b^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立である正方MDS行列による線形変換処理を実行することを特徴とする暗号処理方法にある。

#### 【0018】

さらに、本発明の暗号処理方法の一実施態様において、さらに、前記逆行列： $L_a^{-1}$ 、 $L_b^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が正方MDS行列である正方MDS行列による線形変換処理を実行することを特徴とする。

#### 【0019】

さらに、本発明の暗号処理方法の一実施態様において、前記Feistel型共通鍵ブロック暗号処理のアルゴリズムは、ラウンド数 $2r$ の暗号処理アルゴリズムであり、前記F関数の線形変換処理は、 $r$ 個の全ての偶数ラウンドおよび $r$ 個の全ての奇数ラウンドにおいて $2 \leq q < r$ の $q$ 種類の異なる正方MDS行列を順次繰り返し適用した線形変換処理を実行することを特徴とする。

#### 【0020】

さらに、本発明の暗号処理方法の一実施態様において、前記F関数の線形変換処理において適用する異なる複数の正方MDS行列の各々は、該複数の正方MDS行列を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立である正方MDS行列であることを特徴とする。

#### 【0021】

さらに、本発明の暗号処理方法の一実施態様において、前記F関数の線形変換処理において適用する異なる複数の正方MDS行列の各々は、該複数の正方MDS行列を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列も正方MDS行列となる正方MDS行列であることを特徴とする。

#### 【0022】

さらに、本発明の暗号処理方法の一実施態様において、前記F関数の線形変換処理において適用する異なる複数の正方MDS行列の各々は、該複数の正方MDS行列を構成する要素を全て含む正方MDS行列 $M$ から選択された行ベクトルによって構成される行列 $M'$ から抽出された列ベクトルによって構成される行列によって構成されていることを特徴とする。

#### 【0023】

さらに、本発明の第3の側面は、

Feistel型共通鍵ブロック暗号処理を実行するコンピュータ・プログラムであり、

非線形変換処理および線形変換処理を実行するSPN型のF関数を、複数ラウンド繰り返し実行するステップを有し、

前記複数ラウンド各々に対応するF関数の線形変換処理は、 $m$ 個の非線形変換部各々の出力する $n$ ビット、総計 $mn$ ビットの入力に対する線形変換処理を、正方MDS (Maximum Distance Separable) 行列を適用した線形変換処理として実行する線形変換ステップであり、

前記線形変換ステップにおいて、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々では、異なる正方MDS行列： $L_a$ 、 $L_b$ が適用され、かつ該正方MDS行列の逆行列： $L_a^{-1}$ 、 $L_b^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立である正方MDS行列による線形変換処理を実行することを特徴とするコンピュータ・プログラムにある。



#### 【0024】

なお、本発明のコンピュータ・プログラムは、例えば、様々なプログラム・コードを実行可能なコンピュータ・システムに対して、コンピュータ可読な形式で提供する記憶媒体、通信媒体、例えば、CDやFD、MOなどの記録媒体、あるいは、ネットワークなどの通信媒体によって提供可能なコンピュータ・プログラムである。このようなプログラムをコンピュータ可読な形式で提供することにより、コンピュータ・システム上でプログラムに応じた処理が実現される。

#### 【0025】

本発明のさらに他の目的、特徴や利点は、後述する本発明の実施例や添付する図面に基づくより詳細な説明によって明らかになるであろう。なお、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

#### 【発明の効果】

#### 【0026】

本発明の構成によれば、非線形変換部および線形変換部を有するSPN型のF関数を、複数ラウンド繰り返し実行するFeistel型共通鍵ブロック暗号処理において、複数ラウンド各々に対応するF関数の線形変換処理を、正方MDS(Maximum Distance Separable)行列を適用した線形変換処理として実行するとともに、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列： $L_a$ 、 $L_b$ を適用し、かつ該正方MDS行列の逆行列： $L_a^{-1}$ 、 $L_b^{-1}$ を構成する列ベクトルから任意に選択した $m$ 個の列ベクトルによって構成する行列が線形独立であるか、あるいは正方MDS行列を構成する性質とした正方MDS行列による線形変換処理を実行する構成としたので、共通鍵ブロック暗号における線形攻撃に対する耐性が向上し、暗号鍵等の解析困難性が高まることとなり、安全性の高い暗号処理が実現される。

#### 【0027】

さらに、本発明の構成によれば、非線形変換部および線形変換部を有するSPN型のF関数を、複数ラウンド繰り返し実行するFeistel型共通鍵ブロック暗号処理において、複数ラウンド各々に対応するF関数の線形変換処理を、正方MDS(Maximum Distance Separable)行列を適用した線形変換処理として実行するとともに、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列を適用する構成とし、これらの正方MDS行列自身が、線形独立性を示すか、または正方MDS行列を構成する構成としたので、アクティブSボックスの寄与による同時差分キャンセルの発生しないことが保証され、共通鍵ブロック暗号における差分攻撃に対する強度指標のひとつである暗号化関数全体でのアクティブSボックスの最少数を大きくすることが可能となる。本構成により、線形攻撃、差分攻撃の双方に対して耐性が向上し、より安全性の高い暗号処理が実現される。

#### 【発明を実施するための最良の形態】

#### 【0028】

以下、本発明の暗号処理装置、および暗号処理方法、並びにコンピュータ・プログラムの詳細について説明する。なお、説明は、以下の項目順に行う。

1. 共通鍵ブロック暗号アルゴリズムにおける差分解析処理
2. 共通鍵ブロック暗号アルゴリズムにおける線形解析処理
3. 本発明に基づく暗号処理アルゴリズム

(3-a) 差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例

(3-b) 線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例

(3-c) 差分攻撃および線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例

## 【0029】

### 【1. 共通鍵ブロック暗号アルゴリズムにおける差分解析処理】

まず、DES (Data Encryption Standard) 暗号処理に代表される共通鍵ブロック暗号アルゴリズムにおける差分解析処理の概要について、一般化した共通鍵ブロック暗号モデルを用いて説明する。

## 【0030】

共通鍵ブロック暗号のアルゴリズムは、主として、入力データの変換を実行するラウンド関数部と、ラウンド関数部の各ラウンドで適用する鍵を生成する鍵スケジュール部とに分けることができる。ラウンド関数部の各ラウンドで適用する鍵（副鍵）は、1つのマスター鍵（主鍵）に基づいて、鍵スケジュール部に入力されて生成され、各ラウンド関数部で適用される。この共通鍵暗号方式の代表的な方式に米国連邦標準暗号方式としてのDES (Data Encryption Standard) がある。

## 【0031】

Feistel 構造と呼ばれる代表的な共通鍵ブロック暗号の構造について、図1を参照して説明する。

## 【0032】

Feistel 構造は、変換関数の単純な繰り返しにより、平文を暗号文に変換する構造を持つ。平文の長さを  $2mn$  ビットとする。ただし、 $m$ ,  $n$  は共に整数である。初めに、 $2mn$  ビットの平文を、 $mn$  ビットの2つの入力データ  $P_L$  (Plain-Left) 101,  $P_R$  (Plain-Right) 102 に分割し、これを入力値とする。

## 【0033】

Feistel 構造はラウンド関数とよばれる基本構造の繰り返しで表現され、各ラウンドに含まれるデータ変換関数はF関数120と呼ばれる。図1の構成では、F関数（ラウンド関数）120が  $r$  段繰り返された構成例を示している。

## 【0034】

例えば第1番目のラウンドでは、 $mn$  ビットの入力データ  $X$  と、鍵生成部（図示せず）から入力される  $mn$  ビットのラウンド鍵  $K_1$  103 がF関数120に入力され、F関数120におけるデータ変換処理の後に  $mn$  ビットのデータ  $Y$  を出力する。出力はもう片方の前段からの入力データ（第1段の場合は入力データ  $P_L$ ）と排他的論理和部104において、排他的論理和演算がなされ、 $mn$  ビットの演算結果が次のラウンド関数へと出力される。この処理、すなわちF関数を定められたラウンド数（ $r$ ）だけ繰り返し適用して暗号化処理が完了し、暗号文の分割データ  $C_L$  (Cipher-Left)、 $C_R$  (Cipher-Right) が出力される。以上の構成より、Feistel 構造の復号処理はラウンド鍵を挿入する順序を逆にすることでよく、逆関数を構成する必要がないことが導かれる。

## 【0035】

各ラウンドの関数として設定されるF関数120の構成について、図2を参照して説明する。図2(a)は、1つのラウンドにおけるF関数120に対する入力および出力を示す図であり、図2(b)は、F関数120の構成の詳細を示す図である。F関数120は、図2(b)に示すように、非線形変換層と線形変換層を接続したいわゆるSPN型の構成を有する。

## 【0036】

SPN型のF関数120は、図2(b)に示すように、非線形変換処理を実行する複数のSボックス (S-box) 121を有する。ラウンド関数部の前段からの  $mn$  ビットの入力値  $X$  は、鍵スケジュール部から入力されるラウンド鍵  $K_i$  と排他的論理和が実行され、その出力が  $n$  ビットずつ非線形変換処理を実行する複数（ $m$  個）のSボックス121に入力される。各Sボックスでは、例えば変換テーブルを適用した非線形変換処理が実行される。

## 【0037】

Sボックス121からの出力データである  $mn$  ビットの入力値  $Z$  は、線形変換処理を実行する線形変換部122に入力されて、例えばビット位置の入れ替え処理などの線形変換

処理が実行され、 $m \times n$  ビットの出力値  $Y$  を出力する。この出力値  $Y$  が前段からの入力データと排他的論理和され、次のラウンドの  $F$  関数の入力値とされる。

#### 【0038】

図2に示す  $F$  関数 120 は、入出力ビット長が  $m \times n$  ( $m, n$ : 整数) ビットであり、非線形変換層は  $n$  ビットの入出力を持つ非線形変換層としての  $S$  ボックス 121 は、 $m$  個並列にならんだ構成を有し、線形変換層としての線形変換部 122 は  $n$  次の既約多項式で定義される  $2$  の拡大体  $GF(2^n)$  上の元を要素として持つ  $m$  次の正方行列に基づく線形変換処理を実行する。

#### 【0039】

線形変換部 122 における線形変換処理に適用する正方行列の例を図3に示す。図3に示す正方行列 125 は、 $n=8, m=8$  の場合の例である。非線形変換部 ( $S$  ボックス 121) から出力された  $m$  個の  $n$  ビットデータ  $Z[1], Z[2], \dots, Z[m]$  に対してあらかじめ定められた正方行列 125 を適用した演算により線形変換が施され、 $F$  関数 (ラウンド関数) 出力としての、 $Y[1], Y[2], \dots, Y[m]$  が決定される。ただし、このとき各データの行列の要素に対する線形演算はあらかじめ定められた  $2$  の拡大体  $GF(2^n)$  上で行われる。

#### 【0040】

これまでの  $F$  e i s t e l 型暗号では、すべての段の  $F$  関数に同じ線形変換層を用いているため、差分の伝播時に同時に複数の差分がキャンセルしてしまうという性質が存在した。背景技術の欄において説明したように、暗号解析手法の代表的な手法として、ある差分を持つ入力データ (平文) とその出力データ (暗号文) を多数解析することにより各ラウンド関数における適用鍵を解析する差分解析 (あるいは差分解読法) が知られており、従来の  $D E S$  暗号アルゴリズム等の共通鍵ブロック暗号においては、 $F$  関数 120 部の線形変換部 122 において適用する処理 (変換行列) を、各段のラウンドにおいて等しいものに設定しているため、差分解析が行いやすく、結果として鍵の解析の容易性を招いている。

#### 【0041】

差分の伝播時に、同時に複数の差分がキャンセルする例について、図4を参照して説明する。なお、本明細書においては、差分を表す場合には  $\Delta$  (デルタ) 記号をつけて表す。

#### 【0042】

図4は  $m=8, n=8$  の  $128$  b i t ブロック暗号における3段の同時差分キャンセルの様子を説明する図である。ただし、図中では  $64$  b i t のデータをバイト単位で区切ってベクトルとして表現し、それぞれの要素を  $16$  進数で表記するものとする。

#### 【0043】

3段構成を持つ  $F$  関数での同時差分キャンセルは、例えば以下のデータ状態 1~4 の設定メカニズムに基づいて発生する。以下に説明するメカニズムの発生するデータ状態は、多数の差分入力データを設定することで発生させることができるデータ状態であり、いわゆる差分解析における鍵 (ラウンド鍵) の解析において発生し得る。

#### 【0044】

(状態1)

$i$  ラウンドへの入力差分の左半分は、すべてゼロである入力差分 ( $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$ ) であり、右半分の入力差分がただひとつの  $S$  - b o x への入力を除いてゼロである入力差分 ( $\Delta X_i = (34, 00, 00, 00, 00, 00, 00, 00)$ ) であるとする。このデータ状態は、多数の差分入力データを設定することで、 $i$  ラウンドにおいて、このようなデータ状態を得ることができるということである。

#### 【0045】

なお、 $\Delta X_i = (34, 00, 00, 00, 00, 00, 00, 00)$  の8つの各要素は、 $F$  関数中に構成される  $m$  個 ( $m=8$ ) の  $S$  ボックス各々に対する入力差分に対応する。差分 (34) が第1  $S$  ボックス (図4中の (S1)) に入力され、(00) が、第2~

8 S ボックスに対する入力差分である。

【0046】

なおゼロ(00)の入力差分を持つSボックスの出力差分はゼロ(00)であり、差分データに関する限り、ゼロ(00)の入力差分を持つSボックスは、何の作用も行っていないものであり、アクティブでないすなわち非アクティブSボックスと呼ばれる。一方、非ゼロの入力差分(図4の例では差分:34)を持つSボックスは、非ゼロの入力差分に対応した非線形変換結果を出力差分として発生させるので、アクティブSボックス(Active S-box)と呼ばれる。

【0047】

図4の例では、非ゼロの入力差分(34)を入力する1つのアクティブSボックス(S1)の出力差分(b7)を発生させており、その他の非アクティブSボックスS2~S8はゼロの入力差分(00)に基づいて出力差分(00)を発生させ、線形変換部の差分入力としている。

【0048】

(状態2)

i ラウンドへの非ゼロの入力差分(図4の例では差分:34)を持つSボックス(以下、アクティブSボックス(Active S-box)と呼ぶ)からの出力差分は線形変換層で拡散されたのちF関数から出力(出力値= $\Delta Y_i$ )され、そのまま次のラウンドへの入力差分 $\Delta X_{i+1}$ となる。

【0049】

図4の例における線形変換は、各ラウンドのF関数において共通する例えば図5に示すある特定の正方行列125による線形変換が実行されi ラウンドのF関数出力差分としての $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$ を出力する。図5に示す線形変換構成から理解されるように、出力差分 $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$ は、1つのアクティブSボックス(S1)からの出力要素Z[1]=b7にのみ依存した値として決定される。

【0050】

このi ラウンドのF関数出力差分としての $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$ は、図4に示す排他的論理和部131において、すべてゼロである入力差分( $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$ )と排他的論理和(XOR)演算が実行され、演算結果が、次のラウンド(i+1)への入力差分 $\Delta X_{i+1}$ となる。

【0051】

i ラウンドのF関数出力差分としての $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$ と、すべてゼロである入力差分 $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$ との排他的論理和(XOR)結果は、 $\Delta Y_i$ であるので、次のラウンド(i+1)への入力差分 $\Delta X_{i+1} = \Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$ となる。

【0052】

(状態3)

i+1 ラウンドのF関数からの出力差分 $\Delta Y_{i+1}$ が、i ラウンドでのActive S-boxの位置にのみ非ゼロ値をもつ。このデータ状態は、多数の差分入力データを設定することで、このようなデータ状態を得ることができるということである。

【0053】

すなわち、 $\Delta Y_{i+1} = (ad, 00, 00, 00, 00, 00, 00, 00)$ であり、i ラウンドと同様、非ゼロの差分値(図4の例では差分:34)を持つS-boxの位置(第1Sボックス(S1))にのみ非ゼロ値をもつ。なお、明らかに $ad \neq 00$ である。

【0054】

(状態4)

$i + 2$  ラウンドのアクティブ S ボックス (Active S-box) (S1) の出力差分が  $i$  ラウンドでのアクティブ S ボックス (Active S-box) (S1) の出力差分と一致した場合、すなわち、図 4 に示すように  $i + 2$  ラウンドのアクティブ S ボックス (S1) の出力差分が b7 となり、 $i$  ラウンドでのアクティブ S ボックス (S1) の出力差分 (b7) と一致する。このデータ状態は、多数の差分入力データを設定することで、このようなデータ状態を得ることができるということである。

#### 【0055】

このデータ状態が発生すると、 $i + 2$  ラウンドの F 関数の出力差分  $\Delta Y_{i+2} = (98, c4, b4, d3, ac, 72, 0f, 32)$  が、2 つ前のラウンドである  $i$  ラウンドの F 関数の出力差分  $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$  と一致することになる。

#### 【0056】

この結果、排他的論理和部 133 では、

$\Delta X_{i+1} = \Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$  と、

$\Delta Y_{i+2} = (98, c4, b4, d3, ac, 72, 0f, 32)$  と、

の同一の値同士の排他的論理和演算が実行されることになり、排他的論理和演算結果としてオール 0 の値を出力する。

#### 【0057】

その結果、次の段 (ラウンド  $i + 3$ ) への出力差分の前段 ( $i + 2$  ラウンド) からの左の入力差分  $\Delta X_{i+3} = (00, 00, 00, 00, 00, 00, 00, 00)$  となる。

#### 【0058】

このラウンド  $i + 3$  への左入力  $\Delta X_{i+3} = (00, 00, 00, 00, 00, 00, 00, 00)$  は、ラウンド  $i$  への左入力  $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$  と同様オールゼロであり、ラウンド  $i + 3$  以降のラウンドにおいても、ラウンド  $i \sim i + 2$  と同様の処理が繰り返される可能性がある。

#### 【0059】

この結果、ラウンド数の伸びに対してアクティブ S ボックスの数が増大せず、差分攻撃に対する強度がそれほど伸びないという問題を発生させる。

#### 【0060】

共通鍵ブロック暗号において、差分攻撃に対する強度指標のひとつとして、暗号化関数全体でのアクティブ S ボックスの最少数が知られている。アクティブ S ボックス数の最少数が大きいほど差分攻撃に対する耐性が高いと判断される。

#### 【0061】

前述したように、差分解析 (差分攻撃) においては、ある差分を持つ入力データ (平文) とその出力データ (暗号文) を多数設定してこの対応を解析することにより各ラウンド関数における適用鍵を解析する手法であり、この差分解析において、アクティブ S ボックスの数を少なくできれば、解析が容易となり、解析プロセス数を削減できる。

#### 【0062】

上述の図 4 を参照した例では、第 1 の S ボックス (S1) のみがアクティブ S ボックスであるパターンの発生状態を提示したが、その他の S ボックス (S2 ~ S8) についても、差分解析の入力データの設定によって、各 S ボックスのみをアクティブ S ボックスとした設定が可能であり、このような差分解析プロセスを実行することにより、各 S ボックスの非線形変換処理の解析、さらに F 関数に対して入力されるラウンド鍵の解析が可能となる。

#### 【0063】

このような差分解析に対する耐性を向上させるためには、アクティブ S ボックスの数が常に多い状態を維持すること、すなわち、アクティブ S ボックスの最少数が多いことが必要である。

#### 【0064】

図 4 を参照して説明した例において、右から左へ入力を行なう F 関数、すなわち、第  $i$

ラウンドと第  $i + 2$  ラウンドのみをアクティブ S ボックス算出処理対象ラウンドとしてみた場合、アクティブ S ボックス数はわずか 2 であり、左から右へ入力を行なう F 関数、すなわち、第  $i + 1$  ラウンドではアクティブ S ボックス数が 8 であるものの、同時差分キャンセルにより第  $i + 3$  ラウンドでのアクティブ S ボックス数が 0 となってしまうため、差分解析による各 S ボックスの非線形変換処理の解析処理が容易となる。

#### 【0065】

図 4 に示す共通鍵ブロック暗号アルゴリズムは、各ラウンドにおける線形変換部において適用する線形変換行列が等しいものであり、この構成に起因して、特に右から左へ入力を行う F 関数におけるわずか 2 つのアクティブ S ボックスにより同時差分キャンセルの発生可能性を引き起こしている。従って、ラウンド数の伸びに対してアクティブ S ボックスの最少数が十分に増大せず、差分攻撃に対する強度がそれほど伸びないという問題がある。

#### 【0066】

次に、同様に、同じ線形変換行列をすべての段（ラウンド）の F 関数に用いる構成において、5 ラウンドにまたがる同時差分キャンセルの発生メカニズムについて、図 6 を参照して説明する。

#### 【0067】

図 6 は  $m = 8$  ,  $n = 8$  の 128 bit ブロック暗号における 5 段の同時差分キャンセルの様子を説明する図である。ただし、図中では 64 bit のデータをバイト単位で区切ってベクトルとして表現し、それぞれの要素を 16 進数で表記するものとする。

#### 【0068】

5 段構成を持つ F 関数での同時差分キャンセルは、例えば以下のデータ状態 1 ~ 7 の設定メカニズムに基づいて発生する。以下に説明するメカニズムの発生するデータ状態は、多数の差分入力データを設定することで発生させることができるデータ状態であり、いわゆる差分解析における鍵（ラウンド鍵）の解析において発生し得る。

#### 【0069】

（状態 1）

$i$  ラウンドへの入力差分の左半分は、すべてゼロである入力差分（ $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$ ）であり、右半分の入力差分がただひとつの S-box への入力を除いてゼロである入力差分（ $\Delta X_i = (34, 00, 00, 00, 00, 00, 00, 00)$ ）であるとする。このデータ状態は、多数の差分入力データを設定することで、 $i$  ラウンドにおいて、このようなデータ状態を得ることができるということである。

#### 【0070】

なお、 $\Delta X_i = (34, 00, 00, 00, 00, 00, 00, 00)$  の 8 つの各要素は、F 関数中に構成される  $m$  個（ $m = 8$ ）の S ボックス各々に対する入力差分に対応する。（34）が第 1 S ボックス（図 6 中の（S1））に入力され、（00）が、第 2 ~ 8 S ボックスに対する入力差分である。

#### 【0071】

なお前述したように、ゼロ（00）の入力差分を持つ S ボックスの出力差分はゼロ（00）であり、差分データに関する限り、ゼロ（00）の入力差分を持つ S ボックスは、何の作用も行っていないものであり、アクティブでないすなわち非アクティブ S ボックスと呼ばれる。一方、非ゼロの入力差分（図 6 の例では差分：34）を持つ S ボックス（S1）のみが、非ゼロの入力差分に対応した非線形変換結果を出力差分として発生させるので、アクティブ S ボックス（Active S-box）である。

#### 【0072】

図 6 の例では、非ゼロの入力差分（34）を入力する 1 つのアクティブ S ボックス（S1）の出力差分（b7）を発生させており、その他の非アクティブ S ボックス S2 ~ S8 はゼロの入力差分（00）に基づいて出力差分（00）を発生させ、線形変換部の差分入力としている。

### 【0073】

#### (状態2)

i ラウンドへの非ゼロの入力差分 (図4の例では差分: 34) を持つSボックス (以下、アクティブSボックス (Active S-box) と呼ぶ) からの出力差分は線形変換層で拡散されたのちF関数から出力 (出力値 =  $\Delta Y_i$ ) され、そのまま次のラウンドへの入力差分  $\Delta X_{i+1}$  となる。

### 【0074】

図6の例において、各ラウンドに共通の例えば図5に示すある特定の正方行列 125 による線形変換が実行され i ラウンドのF関数出力差分としての  $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$  を出力する。

### 【0075】

i ラウンドのF関数出力差分としての  $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$  は、図6に示す排他的論理和部 141 において、すべてゼロである入力差分 ( $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$ ) と排他的論理和 (XOR) 演算が実行され、演算結果が、次のラウンド (i+1) への入力差分  $\Delta X_{i+1}$  となる。

### 【0076】

i ラウンドのF関数出力差分としての  $\Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$  と、すべてゼロである入力差分 ( $\Delta X_{i-1} = (00, 00, 00, 00, 00, 00, 00, 00)$ ) との排他的論理和 (XOR) 結果は、 $\Delta Y_i$  であるので、次のラウンド (i+1) への入力差分  $\Delta X_{i+1} = \Delta Y_i = (98, c4, b4, d3, ac, 72, 0f, 32)$  となる。

### 【0077】

#### (状態3)

i+1 ラウンドのF関数からの出力差分  $\Delta Y_{i+1}$  が、i ラウンドでの Active S-box の位置にのみ非ゼロ値をもつ。このデータ状態は、多数の差分入力データを設定することで、このようなデータ状態を得ることができるということである。

### 【0078】

すなわち、 $\Delta Y_{i+1} = (34, 00, 00, 00, 00, 00, 00, 00)$  であり、i ラウンドと同様、非ゼロの差分値 (図6の例では差分: 34) を持つS-box の位置 (第1Sボックス (S1)) にのみ非ゼロ値をもつ。

### 【0079】

#### (状態4)

i+2 ラウンドのF関数に対する入力は、 $\Delta X_i = (34, 00, 00, 00, 00, 00, 00, 00)$  と、 $\Delta Y_{i+1} = (34, 00, 00, 00, 00, 00, 00, 00)$  との排他的論理和部 142 における排他的論理和結果、すなわち、同一データ同士の排他的論理和であり、オールゼロの入力、 $\Delta X_{i+2} = (00, 00, 00, 00, 00, 00, 00, 00)$  となり、その結果、i+2 ラウンドのF関数からの出力差分も、オールゼロの出力差分、 $\Delta Y_{i+2} = (00, 00, 00, 00, 00, 00, 00, 00)$  となる。

### 【0080】

#### (状態5)

i+3 ラウンドのF関数に対する入力は、 $\Delta X_{i+1} = (98, c4, b4, d3, ac, 72, 0f, 32)$  と、オールゼロの i+2 ラウンドのF関数出力差分  $\Delta Y_{i+2} = (00, 00, 00, 00, 00, 00, 00, 00)$  との排他的論理和部 143 における排他的論理和結果であり、i+3 ラウンドのF関数に対する入力  $\Delta X_{i+3} = \Delta X_{i+1} = (98, c4, b4, d3, ac, 72, 0f, 32)$  となる。

### 【0081】

#### (状態6)

i+3 ラウンドのF関数出力差分が、 $\Delta Y_{i+3} = (43, 00, 00, 00, 00, 00,$

0 0 , 0 0 , 0 0 ) となり、オールゼロの  $\Delta X_{i+2} = (0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0)$  との排他的論理和部 1 4 4 における排他的論理和の結果としての  $\Delta X_{i+4} = \Delta Y_{i+3} = (4 3 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0)$  が  $i+4$  ラウンドの F 関数入力差分となる。

#### 【0 0 8 2】

(状態 7)

$i+4$  ラウンドのアクティブ S ボックス (Active S-box) (S1) の出力差分が  $i$  ラウンドでのアクティブ S ボックス (Active S-box) (S1) の出力差分と一致した場合、すなわち、図 6 に示すように  $i+4$  ラウンドのアクティブ S ボックス (S1) の出力差分が b 7 となり、 $i$  ラウンドでのアクティブ S ボックス (S1) の出力差分 (b 7) と一致する。このデータ状態は、多数の差分入力データを設定することで、このようなデータ状態を得ることができるということである。

#### 【0 0 8 3】

このデータ状態が発生すると、 $i+4$  ラウンドの F 関数の出力差分  $\Delta Y_{i+4} = (9 8 , c 4 , b 4 , d 3 , a c , 7 2 , 0 f , 3 2)$  が、2 つ前のラウンドである  $i+2$  ラウンドの排他的論理和部 1 4 3 の出力差分  $\Delta X_{i+3} = (9 8 , c 4 , b 4 , d 3 , a c , 7 2 , 0 f , 3 2)$  と一致することになる。

#### 【0 0 8 4】

この結果、排他的論理和部 1 4 5 では、

$\Delta X_{i+3} = (9 8 , c 4 , b 4 , d 3 , a c , 7 2 , 0 f , 3 2)$  と、

$\Delta Y_{i+4} = (9 8 , c 4 , b 4 , d 3 , a c , 7 2 , 0 f , 3 2)$  と、

の同一の値同士の排他的論理和演算が実行されることになり、排他的論理和演算結果としてオール 0 の値を出力する。

#### 【0 0 8 5】

その結果、次の段 (ラウンド  $i+5$ ) への入力差分は、 $\Delta X_{i+5} = (0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0)$  として設定される。

#### 【0 0 8 6】

このラウンド  $i+5$  への左入力  $\Delta X_{i+5} = (0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0)$  は、ラウンド  $i$  への左入力  $\Delta X_{i-1} = (0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0 , 0 0)$  と同様オールゼロであり、ラウンド  $i+5$  以降のラウンドにおいても、ラウンド  $i \sim i+4$  と同様の処理が繰り返される可能性がある。

#### 【0 0 8 7】

この結果、ラウンド数の伸びに対してアクティブ S ボックスの数が増大せず、差分攻撃に対する強度がそれほど伸びないという問題を発生させる。

#### 【0 0 8 8】

前述したように、差分解析 (差分攻撃) においては、ある差分を持つ入力データ (平文) とその出力データ (暗号文) を多数設定してこの対応を解析することにより各ラウンド関数における適用鍵を解析する手法であり、この差分解析において、アクティブ S ボックスの数を少なくできれば、解析が容易となり、解析プロセス数を削減できる。

#### 【0 0 8 9】

上述の図 6 を参照した例において、右から左へ入力を行なう F 関数、すなわち、第  $i$  ラウンドと第  $i+2$  ラウンド、第  $i+4$  ラウンドのみをアクティブ S ボックス算出処理対象ラウンドとしてみた場合、アクティブ S ボックス数は、第  $i$  ラウンド = 1、第  $i+2$  ラウンド = 0、第  $i+4$  ラウンド = 1 の合計わずか 2 であり、左から右へ入力を行なう F 関数、すなわち第  $i+1$  ラウンドおよび第  $i+3$  ラウンドではアクティブ S ボックス数が 8 であるものの、同時差分キャンセルにより第  $i+5$  ラウンドでのアクティブ S ボックス数が 0 となってしまうため、差分解析による各 S ボックスの非線形変換処理の解析、および、F 関数に対する入力ラウンド鍵の解析処理が比較的、容易となる。

#### 【0 0 9 0】

図 6 を参照した例では、第 1 の S ボックス (S1) のみがアクティブ S ボックスである



パターンの発生状態を提示したが、その他のSボックス（S2～S8）についても、差分解析の入力データの設定によって、各SボックスのみをアクティブSボックスとした設定が可能であり、このような差分解析プロセスを実行することにより、各Sボックスの非線形変換処理の解析、さらにF関数に対して入力されるラウンド鍵の解析が可能となる。

#### 【0091】

図4および図6を参照して、3および5ラウンドの場合の同時差分キャンセルの発生例を説明したが、任意のラウンド数に一般化して同時差分キャンセルを定義すると以下のように定義することができる。図7を参照して、任意のラウンド数における同時差分キャンセルの定義について説明する。なお、図7は、フェイステル（Feistel）構造の共通鍵ブロック暗号を実行するフェイステル（Feistel）構造の1つおきのラウンド（ $i, i+2, i+4, \dots, i+2j$ ）を示している。

#### 【0092】

##### 「定義」

フェイステル（Feistel）構造のラウンド*i*での入力差分の半分（PLまたはPR）が0（図7において、 $\Delta X_i = (00, 00, 00, 00, 00, 00, 00, 00)$ ）であり、そこに*i+2j*ラウンド（ $j=0, 1, 2, \dots$ ）のF関数の出力差分が排他的論理和部で演算されていく過程において、あるラウンド*i+2k*において、排他的論理和の結果が0（図7において、 $\Delta X_{i+2j+1} = (00, 00, 00, 00, 00, 00, 00, 00)$ ）になった場合を“同時差分キャンセル”と呼ぶ。

#### 【0093】

その時、*i, i+2, i+4, \dots, i+2k*ラウンドのF関数の中に存在するアクティブSボックス（Active S-box）のことを“同時差分キャンセルを発生させたアクティブSボックス”と呼ぶものとし、ベクトルAの非ゼロの要素数をハミングウェイト  $hw(A)$  と定義すると、同時差分キャンセルを発生させるアクティブSボックスの数  $a$  は、以下の式として表せる。

#### 【数1】

$$a = \sum_{j=0}^k hw(\Delta X_{i+2j})$$

#### 【0094】

前述の3ラウンド、5ラウンドでの例ではともに同時差分キャンセルを発生させたアクティブSボックス数は2、すなわち  $a=2$  である。

#### 【0095】

前述したように、共通鍵ブロック暗号における差分攻撃に対する強度指標のひとつが暗号化関数全体でのアクティブSボックスの最少数であり、アクティブSボックス数の最少数が大きいほど差分攻撃に対する耐性が高いと判断される。

#### 【0096】

しかし、DESアルゴリズムのように同じ線形変換行列をすべての段のF関数に用いる

構成では、図4、図6を参照して説明したように、わずか2つのアクティブSボックスにより同時差分キャンセルが発生してしまう可能性があった。そのような性質があるためラウンド数の伸びに対してアクティブSボックスの最少数が十分に増大せず、差分攻撃に対する強度がそれほど伸びないという問題があった。

【0097】

〔2．共通鍵ブロック暗号アルゴリズムにおける線形解析処理〕

差分解析処理は、上述したように、解析の実行者が一定の差分を持つ入力データ（平文）を容易し、その対応する出力データ（暗号文）を解析することが必要となる。線形解析処理は、一定の差分を持つ入力データ（平文）を準備する必要はなく、所定量以上の入力データ（平文）と対応する出力データ（暗号文）とに基づいて解析を行う。

【0098】

前述したように、共通鍵ブロック暗号アルゴリズムでは非線形変換部としてのSボックスを有し、入力データ（平文）と対応する出力データ（暗号文）との線形関係はないが、線形解析では、このSボックスの入出力を線形近似し、多数の入力データ（平文）と対応する出力データ（暗号文）の構成ビット値の線形関係を解析し、候補となる鍵を絞り込むことによって解析を行う。線形解析においては、特定の差分を持つ入力データを準備することが必要ではなく、多数の平文と対応暗号文を容易することで、解析が可能となる。

【0099】

〔3．本発明に基づく暗号処理アルゴリズム〕

以下、本発明の暗号処理アルゴリズムについて説明する。本発明の暗号処理アルゴリズムは、上述した線形解析、差分解析等の攻撃に対する耐性を向上させた構成、すなわち、鍵解析の困難性を高め、安全性を向上させた構成を持つ。

【0100】

本発明に係る暗号処理アルゴリズムの1つの特徴は、従来のDESアルゴリズムの如く各ラウンドのF関数に構成される線形変換部に共通の処理（変換行列）を適用した構成とせず、複数の異なる正方MDS（Maximum Distance Separable）行列を設定した構成としたことである。具体的には、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列を適用した線形変換処理を実行する構成を持つ。

【0101】

本発明に係る暗号処理アルゴリズムは、正方MDS（Maximum Distance Separable）行列の性質を利用し、少ないアクティブSボックスに基づく同時差分キャンセルが起こらない、または起こりにくい構造を実現し、アクティブSボックスの最小数を増大させ、差分攻撃に対してより強い共通鍵ブロック暗号処理を実現する。あるいは、既知平文攻撃として行なわれる線形解析についての困難性も高めた構成を持つ。

【0102】

本発明の暗号処理アルゴリズムは、図1、2を参照して説明したSPN型のF関数を有するFeistel構造と呼ばれる代表的な共通鍵ブロック暗号の構造、すなわち、非線形変換部および線形変換部を有するSPN型のF関数の複数ラウンドに渡る単純な繰り返しにより、平文を暗号文に変換する、あるいは暗号文を平文変換する構造を適用している。

【0103】

例えば、平文の長さを $2mn$ ビット（ただし、 $m$ 、 $n$ は共に整数）として、 $2mn$ ビットの平文を、 $mn$ ビットの2つのデータPL（Plain-Left）、PR（Plain-Right）に分割し、これを入力値として、各ラウンドにおいて、F関数を実行させるものであり、F関数は、図2を参照して説明したように、Sボックスからなる非線形変換部と、線形変換部を接続したSPN型を持つF関数である。

【0104】

本発明の構成においては、F関数中の線形変換部において適用する線形変換処理のため

の行列として、複数の異なる正方MDS (Maximum Distance Separable) 行列から選択された行列を各ラウンドのF関数の線形変換部において適用する行列として設定する。具体的には、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列を適用する。

#### 【0105】

正方MDS行列について説明する。正方MDS行列とは以下の(a), (b)の性質を満たす行列をいう。

(a) 正方行列である

(b) 行列に含まれるすべての部分行列 (submatrix) の行列式 (determinant) が0でない、すなわち、 $\det(\text{submatrix}) \neq 0$

#### 【0106】

上記(a), (b)の条件を満足する行列を正方MDS行列と呼ぶ。

共通鍵ブロック暗号の各ラウンドで実行するF関数に対する入出力ビット長が $m \times n$  ( $m, n$ : 整数) ビットであり、F関数内に構成される非線形変換部が $n$ ビットの入出力を持つ $m$ 個のSボックスにより構成され、線形変換部が $n$ 次の既約多項式で定義される2の拡大体 $GF(2^n)$ 上の元を要素として持つ $m$ 次の正方行列に基づく線形変換処理を実行する場合の、正方MDS行列の一例を図8に示す。図8に示す正方MDS行列の例は、 $n=8, m=8$ の正方MDS行列の例である。

#### 【0107】

上記(a), (b)を満足する正方MDS行列は、ベクトルAの非ゼロの要素数をハミングウェイト  $hw(A)$  とし、Mを $m$ 次の正方MDS行列とし、xを正方MDS行列Mへの入力ベクトルとした場合、以下の不等式(式1)を満たすことになる。

$hw(x) + hw(Mx) \geq m + 1 \dots \dots \dots$  (式1)

#### 【0108】

上記式(式1)は、正方MDS行列(M)によって線形変換される入力データxの非ゼロの要素数  $hw(x)$  と、正方MDS行列(M)によって線形変換された出力データ  $Mx$  の非ゼロの要素数  $hw(Mx)$  の総数が、正方MDS行列の次数 $m$ より大となるということを意味している。

#### 【0109】

なお、正方MDS行列という名は正方MDS-code (Maximum Distance Separable Code) の生成行列の標準形の右半分が上記条件を満足していることから名づけているものである。

#### 【0110】

1つの行列をすべてのF関数に組み込むという従来の構成でも線形変換行列に正方MDS行列を用いることで、正方MDS行列でない行列を用いる場合に比べてアクティブSボックス数の最少数を比較的高水準に保持することができるということは知られている。

#### 【0111】

本発明では、各ラウンドのF関数には正方MDS行列の条件を満たす行列を利用し、さらにラウンドごとに異なる行列を設定する方法を提案する。具体的には、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列を適用する。

#### 【0112】

以下に、段数(ラウンド数)が $2r$  ( $r$ は整数)のFeistel型共通鍵ブロック暗号において、差分攻撃に対する耐性をより高めた複数の構成例について説明する。

#### 【0113】

なお、以下の説明において、段数(ラウンド数)が $2r$ のFeistel型共通鍵ブロック暗号処理構成のj段目のF関数における線形変換部で適用する線形変換行列を $MLT_j$ として表すものとする。

#### 【0114】

本発明の構成では、段数(ラウンド数)が $2r$ のFeistel型共通鍵ブロック暗号

処理構成における各段のF関数中の線形変換部において適用する線形変換処理のための行列として、複数の異なる正方MDS (Maximum Distance Separable) 行列から選択された行列を各ラウンドのF関数の線形変換部において適用する行列として設定する。具体的には、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列を適用する。

#### 【0115】

具体的には、段数(ラウンド数)が $2r$ のFeistel型共通鍵ブロック暗号処理構成に対応して、 $r$ 以下の $q$ 個の正方MDS行列： $L_1, L_2, \dots, L_q$ を生成し、段数(ラウンド数)が $2r$ のFeistel型共通鍵ブロック暗号処理構成における奇数段目のF関数中の線形変換部において適用する線形変換処理のための行列として、上位段のF関数から順に $L_1, L_2, \dots, L_q, L_1, L_2, \dots$ として、 $q$ 個の正方MDS行列を繰り返し設定する。さらに、偶数段のF関数については、下位段のF関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$ として、 $q$ 個の正方MDS行列を繰り返し設定する。

#### 【0116】

本設定を適用した構成例を図9に示す。図9は、段数(ラウンド数)が $2r=12$ 、すなわち $r=6$ のFeistel型共通鍵ブロック暗号処理構成とした場合、 $q=3$ 、すなわち、12段のラウンド数を持つFeistel型共通鍵ブロック暗号処理構成において3種類の異なる正方MDS行列を配置した構成例として、各ラウンドのF関数部の線形変換部に設定する正方MDS行列( $L_1, L_2, L_3$ )を示している。

#### 【0117】

図9の構成は、 $2mn$ ビットの平文を、 $mn$ ビットの2つのデータPL (Plain-Left)、PR (Plain-Right)に分割し、これを入力値として、各ラウンドにおいて、F関数を実行させる構成であり、第1ラウンドのF関数401およびその他のラウンドのF関数も、すべて図2を参照して説明したように、Sボックスからなる非線形変換部と、線形変換部を接続したSPN型を持つF関数である。

#### 【0118】

図9の設定例は $r=6$ 、 $q=3$ であり、各F関数内に示す記号 $L_n$ は正方MDS行列402を示している。すなわち、 $L_1, L_2, L_3$ は、それぞれ異なる3種類の正方MDS行列を示し、各F関数の線形変換部において線形変換処理に適用する正方MDS行列を示している。

#### 【0119】

線形変換行列 $MLT_j$ の設定処理シーケンスについて、図10を参照して説明する。

#### 【0120】

##### 【ステップS21】

ラウンド数 $2r$ の半数 $r$ に対して $r$ 以下の数 $q$ 、すなわち、 $q \leq r$ となる数 $q$ を選択する。ただし、 $q$ は2以上の整数である。

##### 【ステップS22】

$q$ 個の $GF(2^n)$ 上の $m$ 次正方MDS行列 $L_1, L_2, \dots, L_q$ を生成する。 $q$ 個の $GF(2^n)$ 上の $m$ 次正方MDS行列 $L_1, L_2, \dots, L_q$ の生成処理手法についての詳細は、後段で説明する。

#### 【0121】

ステップS22において、 $q$ 個の $GF(2^n)$ 上の $m$ 次正方MDS行列 $L_1, L_2, \dots, L_q$ が生成した後、次に、以下の正方MDS行列設定処理を実行する。

##### 【ステップS23】

$2i-1$  ( $1 \leq i \leq r$ ) 段目の線形変換行列 $MLT_{2i-1}$ に $L_{(i-1 \bmod q) + 1}$ を設定する。

##### 【ステップS24】

$2i$  ( $1 \leq i \leq r$ ) 段目の線形変換行列に $MLT_{2i}$ に $MLT_{2r-2i+1}$ を設定する。

### 【0122】

例えば、図9に示す構成例、すなわち、12段( $r=6$ )であり $q=3$ とした場合は、  
MLT1=L1, MLT2=L3  
MLT3=L2, MLT4=L2  
MLT5=L3, MLT6=L1  
MLT7=L1, MLT8=L3  
MLT9=L2, MLT10=L2  
MLT11=L3, MLT12=L1  
の設定となる。

### 【0123】

このように、本発明の暗号処理装置においては、段数(ラウンド数)が $2r$ のFeistel型共通鍵ブロック暗号処理構成に対応して、 $r$ 以下の $q$ 個の正方MDS行列： $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ を生成し、奇数段目については上位段のF関数から順に $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ ， $L_1$ ， $L_2$ ...として、 $q$ 個の正方MDS行列を繰り返し設定し、偶数段のF関数については、下位段のF関数から順に、 $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ ， $L_1$ ， $L_2$ ...として、 $q$ 個の正方MDS行列を繰り返し設定する構成としている。

### 【0124】

次に、図10の処理フローにおけるステップS22の $q$ 個の $GF(2^n)$ 上の $m$ 次正方MDS行列 $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ の生成処理およびF関数への設定構成の詳細について説明する。なお、説明は以下の項目に沿って行なう。

(3-a) 差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例

(3-b) 線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例

(3-c) 差分攻撃および線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例

### 【0125】

〔(3-a) 差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例〕

まず、差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例として、3つの処理例a1，a2，a3について説明する。

### 【0126】

(処理例a1)

差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例の第1の例について説明する。まず、図11に示すフローチャートを参照して正方MDS行列の生成処理について説明する。

### 【0127】

〔ステップS101〕

入力：必要な正方MDSの個数 $q$ ，拡大次数： $n$ ，行列のサイズ： $m$ として、

$GF(2^n)$ 上で、 $q$ 個の $m$ 次正方MDS行列 $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ をランダムに生成する。なお、図11に示すフローでは、MDSの個数 $q=6$ ，拡大次数： $n=8$ ，行列のサイズ： $m=8$ の場合の処理例として示してある。

### 【0128】

〔ステップS102〕

$q$ 個の $m$ 次正方MDS行列 $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ に含まれる $qm$ 個の列の任意の $m$ 個を取り出したときに、線形独立になっているかどうかをチェックする。チェックに通過したらステップS103に進む、そうでない場合はステップS101にもどる。

〔ステップS103〕

$q$ 個の $m$ 次正方MDS行列 $L_1$ ， $L_2$ ， $\dots$ ， $L_q$ を、ラウンド数 $2r$ のFeistel型共通鍵ブロック暗号に適用する正方MDS行列として出力する。

#### 【0129】

以上のプロセスによって、 $q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  が生成される。なお、 $q \leq r$  である。

#### 【0130】

このようにして生成した  $q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  を、先に、図10を参照して説明した、【ステップS23】、【ステップS24】の処理に従って、段数（ラウンド数）が  $2r$  の Feistel 型共通鍵ブロック暗号処理構成の各段のF関数部の線形変換部の線形変換処理に適用する行列として設定する。すなわち、奇数段目については上位段から順に  $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として  $q$  個の正方MDS行列を繰り返し設定し、偶数段のF関数については、下位段のF関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として、 $q$  個の正方MDS行列を繰り返し設定する。

#### 【0131】

このように、偶数ラウンドの正方MDS行列と奇数ラウンドの正方MDS行列を互いに逆順に配置することによって、暗号化処理と復号処理は鍵の順序を入れ替える処理を除き同一であることが保証される。

#### 【0132】

本構成により、

（a）各F関数の線形変換行列は正方MDSであること、

（b）暗号化関数内の奇数ラウンド内の少なくとも連続する  $q$  個のF関数に含まれる線形変換行列の任意の  $m$  個の列ベクトルが独立であること、

（c）偶数ラウンド内の少なくとも連続する  $q$  個のF関数に含まれる線形変換行列の任意の  $m$  個の列ベクトルが独立であること、

これら（a）～（c）が保証されるため、複数段のラウンド数を持つ Feistel 型共通鍵ブロック暗号処理構成において、連続する  $2q-1$  ラウンドにおいて、 $m$  個以下のアクティブSボックスの寄与による同時差分キャンセルは発生しないことが保証される。よって暗号化関数全体のアクティブSボックス数の最小値が増大する。

#### 【0133】

このように、本処理例によって、共通鍵ブロック暗号における差分攻撃に対する強度指標のひとつである暗号化関数全体でのアクティブSボックスの最少数を大きくすることが可能となり、結果として、差分解析（差分攻撃）を行なった場合のアクティブSボックスの数が増大し、解析の困難性が高まることになる。従って鍵の解析の困難な安全性の高い暗号処理が実現される。

#### 【0134】

（処理例a2）

差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例の第2の例について説明する。図12のフローチャートを参照して正方MDS行列の生成処理について説明する。

#### 【0135】

【ステップS201】

入力：必要なMDSの個数  $q$ ，拡大次数： $n$ ，行列のサイズ： $m$ として、

$GF(2^n)$  上で、 $q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  をランダムに生成する。なお、図12に示すフローでは、MDSの個数  $q=6$ ，拡大次数： $n=8$ ，行列のサイズ： $m=8$  の場合の処理例として示してある。

#### 【0136】

【ステップS202】

$q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  に含まれる  $qm$  個の列の任意の  $m$  個を取り出したときに、正方MDS行列になっているかどうかをチェックする。チェックに通過したらステップS203に進む、そうでない場合はステップS201にもどる。

なお、正方MDS行列とは、前述したように以下の性質を満たす行列をいう。

（a）正方行列である

(b) 行列に含まれるすべての部分行列 (submatrix) の行列式 (determinant) が 0 でない、すなわち、 $\det(\text{submatrix}) \neq 0$

【ステップ S 2 0 3】

q 個の m 次正方 MDS 行列  $L_1, L_2, \dots, L_q$  を、ラウンド数  $2r$  の Feistel 型共通鍵ブロック暗号に適用する正方 MDS 行列として出力する。

【0 1 3 7】

以上のプロセスによって、q 個の m 次正方 MDS 行列  $L_1, L_2, \dots, L_q$  が生成される。なお、 $q \leq r$  である。

【0 1 3 8】

前述の処理例 a 1 における正方 MDS 行列生成処理においては、図 1 1 の処理シーケンスにおいて説明したように、ステップ S 1 0 2 において、q 個の m 次正方 MDS 行列  $L_1, L_2, \dots, L_q$  に含まれる qm 個の列の任意の m 個を取り出したときの線形独立性を判定したが、この処理例 a 2 における正方 MDS 行列生成処理においては、q 個の m 次正方 MDS 行列  $L_1, L_2, \dots, L_q$  に含まれる qm 個の列の任意の m 個を取り出したとき正方 MDS 行列になっているかどうかをチェックする。すなわち、より厳しいチェックが実行されることになる。

【0 1 3 9】

この図 1 2 に示す処理シーケンスに従った正方 MDS 行列生成処理によって生成された q 個の m 次正方 MDS 行列  $L_1, L_2, \dots, L_q$  が、先に説明した処理例 a 1 と同様、先に、図 1 0 を参照して説明した、【ステップ S 2 3】、【ステップ S 2 4】の処理に従って、段数 (ラウンド数) が  $2r$  の Feistel 型共通鍵ブロック暗号処理構成の各段の F 関数部の線形変換部の線形変換処理に適用する行列として設定される。すなわち、奇数段目については上位段から順に  $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として q 個の正方 MDS 行列を繰り返し設定し、偶数段の F 関数については、下位段の F 関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として、q 個の正方 MDS 行列を繰り返し設定する。

【0 1 4 0】

このように、偶数ラウンドの正方 MDS 行列と奇数ラウンドの正方 MDS 行列を互いに逆順に配置することによって、暗号化処理と復号処理は鍵の順序を入れ替える処理を除き同一であることが保証される。

【0 1 4 1】

本構成により、

(a) 各 F 関数の線形変換行列は正方 MDS であること、

(b) 暗号化関数内の奇数ラウンド内の少なくとも連続する q 個の F 関数に含まれる線形変換行列の任意の m 個の列ベクトルが正方 MDS 行列であること、

(c) 偶数ラウンド内の少なくとも連続する q 個の F 関数に含まれる線形変換行列の任意の m 個の列ベクトルが正方 MDS 行列であること、

これら (a) ~ (c) が保証されるため、複数段のラウンド数を持つ Feistel 型共通鍵ブロック暗号処理構成において、連続する  $2q - 1$  ラウンドにおいて、m 個以下のアクティブ S ボックスの寄与による同時差分キャンセルは発生しないことが保証される。

さらに、

(d) 正方 MDS の性質から、a 個 ( $a \leq m$ ) のアクティブ S ボックスの寄与によって得られる差分値における非ゼロの要素数は  $m + 1 - a$  個以上になることが保証される。よって暗号化関数全体のアクティブ S ボックス数の最小値が増大する。

【0 1 4 2】

このように、本処理例によって、共通鍵ブロック暗号における差分攻撃に対する強度指標のひとつである暗号化関数全体でのアクティブ S ボックスの最少数を大きくすることが可能となり、結果として、差分解析 (差分攻撃) を行なった場合のアクティブ S ボックスの数が増大し、解析の困難性が高まることになる。従って鍵の解析の困難な安全性の高い暗号処理が実現される。

### 【0143】

(処理例 a3)

差分攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例の第3の例について説明する。図13のフローチャートを参照して正方MDS行列の生成処理について説明する。

### 【0144】

【ステップS301】

入力：必要なMDSの個数 $q$ ，拡大次数： $n$ ，行列のサイズ： $m$ として、

$GF(2^n)$ 上で、1個の $qm$ 次正方MDS行列 $M$ を生成する。なお、図13に示すフローでは、MDSの個数 $q=6$ ，拡大次数： $n=8$ ，行列のサイズ： $m=8$ の場合の処理例として示してある。

### 【0145】

【ステップS302】

1個の $qm$ 次正方MDS行列 $M$ から $m$ 本の行を任意に選択抽出し、 $m$ 行、 $qm$ 列の行列 $M'$ を構成する。

【ステップS303】

$m$ 行、 $qm$ 列の行列 $M'$ に含まれる $qm$ 本の列ベクトルを重複することなく $m$ 本の列ベクトルからなる $q$ 個のグループに任意に分割し、それぞれのグループに含まれる列ベクトルから $m$ 次の正方行列 $L_1, L_2, \dots, L_q$ を、ラウンド数 $2r$ のFeistel型共通鍵ブロック暗号に適用する正方MDS行列として出力する。

### 【0146】

以上のプロセスによって、 $q$ 個の $m$ 次正方MDS行列 $L_1, L_2, \dots, L_q$ が生成される。なお、 $q \leq r$ である。

### 【0147】

処理例 a3における正方MDS行列生成手法3について、図14を参照して、より具体的に説明する。

【ステップS301】

$GF(2^n)$ 上で、1個の $qm$ 次正方MDS行列 $M$ を生成する。図14に示すように、 $qm \times qm$ の正方MDS行列 $M$ を生成する。なお、このステップS301において生成する行列 $M$ の次数は $qm$ 次より大きいものでもよい。

【ステップS302】

図14に示すように、 $qm$ 次正方MDS行列 $M$ から $m$ 本の行を任意に選択抽出し、 $m$ 行、 $qm$ 列の行列 $M'$ を構成する。なお、図に示す例では、連続する $m$ 本の行を選択抽出した例として示してあるが、 $m$ 次正方MDS行列 $M$ を構成する任意の離間した行を $m$ 本選択抽出して、 $m$ 行、 $qm$ 列の行列 $M'$ を構成してもよい。

【ステップS303】

$m$ 行、 $qm$ 列の行列 $M'$ に含まれる $qm$ 本の列ベクトルを重複することなく $m$ 本の列ベクトルからなる $x$ 個のグループに任意に分割し、それぞれのグループに含まれる列ベクトルから $m$ 次の正方行列 $L_1, L_2, \dots, L_x$ を生成する。

### 【0148】

図13、図14を参照して説明した処理シーケンスに従った正方MDS行列生成処理によって生成された $q$ 個の $m$ 次正方MDS行列 $L_1, L_2, \dots, L_q$ が、先に説明した処理例 a1、a2と同様、先に、図10を参照して説明した、【ステップS23】、【ステップS24】の処理に従って、段数（ラウンド数）が $2r$ のFeistel型共通鍵ブロック暗号処理構成の各段のF関数部の線形変換部の線形変換処理に適用する行列として設定される。すなわち、奇数段目については上位段から順に $L_1, L_2, \dots, L_q, L_1, L_2, \dots$ として $q$ 個の正方MDS行列を繰り返し設定し、偶数段のF関数については、下位段のF関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$ として、 $q$ 個の正方MDS行列を繰り返し設定する。

### 【0149】



このように、偶数ラウンドの正方MDS行列と奇数ラウンドの正方MDS行列を互いに逆順に配置することによって、暗号化処理と復号処理は鍵の順序を入れ替える処理を除き同一であることが保証される。

#### 【0150】

本構成により、

(a) 各F関数の線形変換行列は正方MDSであること、

(b) 暗号化関数内の奇数ラウンド内の少なくとも連続する $q$ 個のF関数に含まれる線形変換行列の任意の $m$ 個の列ベクトルが独立であること、

(c) 偶数ラウンド内の少なくとも連続する $q$ 個のF関数に含まれる線形変換行列の任意の $m$ 個の列ベクトルが独立であること、

これら(a)～(c)が保証されるため、複数段のラウンド数を持つFeistel型共通鍵ブロック暗号処理構成において、連続する $2q-1$ ラウンドにおいて、 $m$ 個以下のアクティブSボックスの寄与による同時差分キャンセルは発生しないことが保証される。

さらに、

(d) 正方MDSの性質から、 $a$ 個( $a \leq m$ )のアクティブSボックスの寄与によって得られる差分値における非ゼロの要素数は $m+1-a$ 個以上になることが保証される。よって暗号化関数全体のアクティブSボックス数の最小値が増大する。

#### 【0151】

なお、処理例a3が特に効果を発揮するのは、 $m$ ,  $r$ , が大きくなり、前述した処理例a1, a2の行列決定処理方式にかかる時間的コストが莫大となり、現実的な時間内に行列を決定することが困難である場合である。そのような場合でも本処理例a3の正方MDS行列生成手法ならば比較的短時間での行列生成処理が可能となる。

#### 【0152】

これは、処理例a3においては、大きな $m$ ,  $r$ に対しても現実的な時間で十分に処理可能な方式、例えばリードソロモン(Reed-Solomon)符号の生成行列の生成法を適用することが可能となるからである。

#### 【0153】

この処理例a3においても、上述したように、共通鍵ブロック暗号における差分攻撃に対する強度指標のひとつである暗号化関数全体でのアクティブSボックスの最少数を大きくすることが可能となり、結果として、差分解析(差分攻撃)を行なった場合のアクティブSボックスの数が増大し、解析の困難性が高まることになる。従って鍵の解析の困難な安全性の高い暗号処理が実現される。

#### 【0154】

[(3-b) 線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例]

次に、線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例として、2つの処理例b1, b2について説明する。

#### 【0155】

(処理例b1)

線形攻撃に対する耐性向上を実現した正方MDS行列の生成およびF関数への設定例の第1の例について、説明する。図15に示すフローチャートを参照して正方MDS行列の生成処理について説明する。

#### 【0156】

[ステップS401]

入力：必要な正方MDSの個数 $q$ , 拡大次数： $n$ , 行列のサイズ： $m$ として、

$GF(2^n)$ 上で、 $q$ 個の $m$ 次正方MDS行列 $M_1, M_2, \dots, M_q$ をランダムに生成する。なお、図14に示すフローでは、正方MDSの個数 $q=6$ , 拡大次数： $n=8$ , 行列のサイズ： $m=8$ の場合の処理例として示してある。

#### 【0157】

[ステップS402]

$q$  個の  $m$  次正方 MDS 行列  $M_1, M_2, \dots, M_q$  の逆行列  $M_1^{-1}, M_2^{-1}, \dots, M_q^{-1}$  を算出し、隣り合う 2 つの逆行列に含まれる  $2m$  の行ベクトルから任意の  $m$  本の行ベクトルを取り出したときに、線形独立になっているかどうかをチェックする。図 15 中、 ${}^tR$  は、行ベクトルの転置ベクトルを示すものである。チェックに通過したらステップ S403 に進む、そうでない場合はステップ S401 にもどる。ただし、 $M_1^{-1}$  と  $M_q^{-1}$  は、隣り合う行列とする。

【ステップ S403】

$q$  個の  $m$  次正方 MDS 行列  $L_1, L_2, \dots, L_q$  を、ラウンド数  $2r$  の Feistel 型共通鍵ブロック暗号に適用する正方 MDS 行列として出力する。

【0158】

以上のプロセスによって、 $q$  個の  $m$  次正方 MDS 行列  $L_1, L_2, \dots, L_q$  が生成される。なお、 $q \leq r$  である。

【0159】

このようにして生成した  $q$  個の  $m$  次正方 MDS 行列  $L_1, L_2, \dots, L_q$  を、先に、図 10 を参照して説明した、【ステップ S23】、【ステップ S24】の処理に従って、段数（ラウンド数）が  $2r$  の Feistel 型共通鍵ブロック暗号処理構成の各段の F 関数部の線形変換部の線形変換処理に適用する行列として設定する。すなわち、奇数段目については上位段から順に  $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として  $q$  個の正方 MDS 行列を繰り返し設定し、偶数段の F 関数については、下位段の F 関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として、 $q$  個の正方 MDS 行列を繰り返し設定する。

【0160】

このように、偶数ラウンドの正方 MDS 行列と奇数ラウンドの正方 MDS 行列を互いに逆順に配置することによって、暗号化処理と復号処理は鍵の順序を入れ替える処理を除き同一であることが保証される。

【0161】

本構成により、

（a）各 F 関数の線形変換行列は正方 MDS であること、

（b）暗号化関数内の奇数ラウンド内に連続して含まれる線形変換行列、および偶数ラウンド内に連続して含まれる線形変換行列の逆行列の任意の  $m$  個の列ベクトルは独立であること、

が保証される。このことにより、線形攻撃における線形近似による解析困難性を高めることが可能となり、解析の困難性、すなわち鍵の解析の困難な安全性の高い暗号処理が実現される。

【0162】

（処理例 b2）

線形攻撃に対する耐性向上を実現した正方 MDS 行列の生成および F 関数への設定例の第 2 の例について、説明する。図 16 に示すフローチャートを参照して正方 MDS 行列の生成処理について説明する。

【0163】

【ステップ S501】

入力：必要な正方 MDS の個数  $q$ ，拡大次数： $n$ ，行列のサイズ： $m$  として、

$GF(2^n)$  上で、 $q$  個の  $m$  次正方 MDS 行列  $M_1, M_2, \dots, M_q$  をランダムに生成する。なお、図 16 に示すフローでは、正方 MDS の個数  $q = 6$ ，拡大次数： $n = 8$ ，行列のサイズ： $m = 8$  の場合の処理例として示してある。

【0164】

【ステップ S502】

$q$  個の  $m$  次正方 MDS 行列  $M_1, M_2, \dots, M_q$  の逆行列  $M_1^{-1}, M_2^{-1}, \dots, M_q^{-1}$  を算出し、隣り合う 2 つの逆行列に含まれる  $2m$  の行ベクトルから任意の  $m$  本の行ベクトルを取り出したときに、正方 MDS 行列になっているかどうかをチェックする。図 16 中、 ${}^tR$  は、行ベクトルの転置ベクトルを示すものである。チェックに通過したら

ステップS503に進む、そうでない場合はステップS401にもどる。ただし、 $M_{1-1}$ と $M_{q-1}$ は、隣り合う行列とする。

なお、正方MDS行列とは、前述したように以下の性質を満たす行列をいう。

(a) 正方行列である

(b) 行列に含まれるすべての部分行列 (submatrix) の行列式 (determinant) が0でない、すなわち、 $\det(\text{submatrix}) \neq 0$

【0165】

【ステップS503】

$q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  を、ラウンド数  $2r$  の Feistel 型共通鍵ブロック暗号に適用する正方MDS行列として出力する。

【0166】

以上のプロセスによって、 $q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  が生成される。なお、 $q \leq r$  である。

【0167】

前述の処理例 b1 における正方MDS行列生成処理においては、図15の処理シーケンスにおいて説明したように、ステップS402において、 $q$  個の  $m$  次正方MDS行列の  $M_1, M_2, \dots, M_q$  の逆行列  $M_{1-1}, M_{2-1}, \dots, M_{q-1}$  に含まれる  $qm$  個の列の任意の  $m$  個を取り出したときの線形独立性を判定したが、この処理例 b2 における正方MDS行列生成処理においては、 $q$  個の  $m$  次正方MDS行列の  $M_1, M_2, \dots, M_q$  の逆行列  $M_{1-1}, M_{2-1}, \dots, M_{q-1}$  に含まれる  $qm$  個の列の任意の  $m$  個を取り出したとき正方MDS行列になっているかどうかをチェックする。すなわち、より厳しいチェックが実行されることになる。

【0168】

この図16に示す処理シーケンスに従った正方MDS行列生成処理によって生成された  $q$  個の  $m$  次正方MDS行列  $L_1, L_2, \dots, L_q$  が、先に説明した処理例 b1 と同様、先に、図10を参照して説明した、【ステップS23】、【ステップS24】の処理に従って、段数 (ラウンド数) が  $2r$  の Feistel 型共通鍵ブロック暗号処理構成の各段の F 関数部の線形変換部の線形変換処理に適用する行列として設定される。すなわち、奇数段目については上位段から順に  $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として  $q$  個の正方MDS行列を繰り返し設定し、偶数段の F 関数については、下位段の F 関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として、 $q$  個の正方MDS行列を繰り返し設定する。

【0169】

このように、偶数ラウンドの正方MDS行列と奇数ラウンドの正方MDS行列を互いに逆順に配置することによって、暗号化処理と復号処理は鍵の順序を入れ替える処理を除き同一であることが保証される。

【0170】

本構成により、

(a) 各 F 関数の線形変換行列は正方MDSであること、

(b) 暗号化関数内の奇数ラウンド内に連続して含まれる線形変換行列、および偶数ラウンド内に連続して含まれる線形変換行列の逆行列の任意の  $m$  個の列ベクトルが正方MDS行列となること、

が保証される。このことにより、線形攻撃における線形近似による解析困難性を高めることが可能となり、解析の困難性、すなわち鍵の解析の困難な安全性の高い暗号処理が実現される。

【0171】

【(3-c) 差分攻撃および線形攻撃に対する耐性向上を実現した正方MDS行列の生成および F 関数への設定例】

次に、差分攻撃および線形攻撃に対する耐性向上を実現した正方MDS行列の生成および F 関数への設定例について説明する。

#### 【0172】

差分攻撃に対する耐性を持つ暗号処理アルゴリズムは、先に、図10～図13を参照して説明した処理、すなわち、F関数の線形処理部における線形変換に適用する正方MDS行列を前述の処理例a1（図11）～a3（図13）のいずれかの処理を適用して設定することで実現される。また、線形攻撃に対する耐性を持つ暗号処理アルゴリズムは、先に、図10、および図14、図15を参照して説明した処理、すなわち、F関数の線形処理部における線形変換に適用する正方MDS行列を前述の処理例b1（図14）、b2（図15）のいずれかの処理を適用して設定することで実現される。

#### 【0173】

差分攻撃および線形攻撃に対する耐性向上を実現した正方MDS行列は、

処理例a1（図11）～a3（図13）のいずれかの処理と、

処理例b1（図14）、b2（図15）のいずれかの処理とを、

併せて実行して生成した正方MDS行列を、図10において説明した【ステップS23】、【ステップS24】の処理に従って、段数（ラウンド数）が $2r$ のFeistel型共通鍵ブロック暗号処理構成の各段のF関数部の線形変換部の線形変換処理に適用する行列として設定することで実現される。

#### 【0174】

すなわち、

処理例a1と処理例b1、

処理例a1と処理例b2、

処理例a2と処理例b1、

処理例a2と処理例b2、

処理例a3と処理例b1、

処理例a3と処理例b2、

のいずれかの組み合わせによって、 $q$ 個の正方MDS行列を生成し、 $2r$ のFeistel型共通鍵ブロック暗号処理構成の各段のF関数部の線形変換部の線形変換処理に適用する行列として設定する。奇数段目については上位段から順に $L1, L2, \dots, Lq, L1, L2 \dots$ として $q$ 個の正方MDS行列を繰り返し設定し、偶数段のF関数については、下位段のF関数から順に、 $L1, L2, \dots, Lq, L1, L2 \dots$ として、 $q$ 個の正方MDS行列を繰り返し設定する。この設定により、差分攻撃および線形攻撃に対する耐性向上を実現した暗号処理が可能となる。

#### 【0175】

図17を参照して、差分攻撃および線形攻撃に対する耐性向上を実現した暗号処理を実現するための正方MDS行列の生成処理の一例について説明する。この処理は、前述した処理例a2と、処理例b2との組み合わせである。

#### 【0176】

##### 【ステップS601】

入力：必要な正方MDSの個数 $q$ ，拡大次数： $n$ ，行列のサイズ： $m$ として、

$GF(2^n)$ 上で、 $q$ 個の $m$ 次正方MDS行列 $M1, M2, \dots, Mq$ をランダムに生成する。なお、図17に示すフローでは、正方MDSの個数 $q=6$ ，拡大次数： $n=8$ ，行列のサイズ： $m=8$ の場合の処理例として示してある。

#### 【0177】

##### 【ステップS602】

$q$ 個の $m$ 次正方MDS行列 $M1, M2, \dots, Mq$ に含まれる $qm$ 個の列の任意の $m$ 個を取り出したときに、正方MDS行列になっているかどうかをチェックする。チェックに通過したらステップS603に進む、そうでない場合はステップS601にもどる。

なお、正方MDS行列とは、前述したように以下の性質を満たす行列をいう。

(a) 正方行列である

(b) 行列に含まれるすべての部分行列(submatrix)の行列式(determinant)が0でない、すなわち、 $\det(\text{submatrix}) \neq 0$

【0178】

【ステップS603】

$q$  個の  $m$  次正方 MDS 行列  $M_1, M_2, \dots, M_q$  の逆行列  $M_1^{-1}, M_2^{-1}, \dots, M_q^{-1}$  を算出し、隣り合う 2 つの逆行列に含まれる  $2m$  の行ベクトルから任意の  $m$  本の行ベクトルを取り出したときに、正方 MDS 行列になっているかどうかをチェックする。図 17 中、 $tR$  は、行ベクトルの転置ベクトルを示すものである。チェックに通過したらステップ S604 に進む、そうでない場合はステップ S601 にもどる。ただし、 $M_1^{-1}$  と  $M_q^{-1}$  は、隣り合う行列とする。

【0179】

【ステップS604】

$q$  個の  $m$  次正方 MDS 行列  $L_1, L_2, \dots, L_q$  を、ラウンド数  $2r$  の Feistel 型共通鍵ブロック暗号に適用する正方 MDS 行列として出力する。

【0180】

以上のプロセスによって、 $q$  個の  $m$  次正方 MDS 行列  $L_1, L_2, \dots, L_q$  が生成される。なお、 $q \leq r$  である。

【0181】

この図 17 に示す処理シーケンスに従った正方 MDS 行列生成処理によって生成された  $q$  個の  $m$  次正方 MDS 行列  $L_1, L_2, \dots, L_q$  が、先に、図 10 を参照して説明した、【ステップS23】、【ステップS24】の処理に従って、段数（ラウンド数）が  $2r$  の Feistel 型共通鍵ブロック暗号処理構成の各段の F 関数部の線形変換部の線形変換処理に適用する行列として設定される。すなわち、奇数段目については上位段から順に  $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として  $q$  個の正方 MDS 行列を繰り返し設定し、偶数段の F 関数については、下位段の F 関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$  として、 $q$  個の正方 MDS 行列を繰り返し設定する。

【0182】

このように、偶数ラウンドの正方 MDS 行列と奇数ラウンドの正方 MDS 行列を互いに逆順に配置することによって、暗号化処理と復号処理は鍵の順序を入れ替える処理を除き同一であることが保証される。

【0183】

本構成により、

（a）各 F 関数の線形変換行列は正方 MDS であること、

（b）暗号化関数内の奇数ラウンド内の少なくとも連続する  $q$  個の F 関数に含まれる線形変換行列の任意の  $m$  個の列ベクトルが正方 MDS 行列であること、

（c）偶数ラウンド内の少なくとも連続する  $q$  個の F 関数に含まれる線形変換行列の任意の  $m$  個の列ベクトルが正方 MDS 行列であること、

これら（a）～（c）が保証されるため、複数段のラウンド数を持つ Feistel 型共通鍵ブロック暗号処理構成において、連続する  $2q-1$  ラウンドにおいて、 $m$  個以下のアクティブ S ボックスの寄与による同時差分キャンセルは発生しないことが保証される。

さらに、

（d）正方 MDS の性質から、 $a$  個（ $a \leq m$ ）のアクティブ S ボックスの寄与によって得られる差分値における非ゼロの要素数は  $m+1-a$  個以上になることが保証される。よって暗号化関数全体のアクティブ S ボックス数の最小値が増大する。

さらに、

（e）暗号化関数内の奇数ラウンド内に連続して含まれる線形変換行列、および偶数ラウンド内に連続して含まれる線形変換行列の逆行列の任意の  $m$  個の列ベクトルが正方 MDS 行列となることが保証される。このことにより、線形攻撃における線形近似による解析困難性を高めることが可能となり、解析の困難性、すなわち鍵の解析の困難な安全性の高い暗号処理が実現される。

【0184】

このように、本処理例によって、差分攻撃および線形攻撃の双方の解析の困難性が向上

し、鍵の解析の困難な安全性の高い暗号処理が実現される。なお、図17に示した例は、前述したように、先に説明した処理例a2と処理例b2の組み合わせによる正方MDS行列の生成例であるが、その他、処理例a1と処理例b1、処理例a1と処理例b2、処理例a2と処理例b1、処理例a3と処理例b1、処理例a3と処理例b2とを組み合わせることでq個の正方MDS行列の生成を行い、段数（ラウンド数）が2rのFeistel型共通鍵ブロック暗号処理構成の各段のF関数部の線形変換部の線形変換処理に適用する行列として、奇数段目については上位段から順に $L_1, L_2, \dots, L_q, L_1, L_2, \dots$ としてq個の正方MDS行列を繰り返し設定し、偶数段のF関数については、下位段のF関数から順に、 $L_1, L_2, \dots, L_q, L_1, L_2, \dots$ として、q個の正方MDS行列を繰り返し設定することによっても、差分攻撃および線形攻撃の双方の解析の困難性が高く、鍵の解析の困難な安全性の高い暗号処理が実現可能である。

#### 【0185】

これまでの説明では、分かりやすさを優先して線形変換行列を $GF(2^n)$ 上で定義される $m \times m$ の行列として、 $mn$ ビットから $mn$ ビットへのデータ変換演算としてきたが、差分解読および線形解析に対する同様な効果が $GF(2)$ 上で定義される $mn \times mn$ の行列を用いた場合でも有効である。実際、任意の $GF(2^n)$ 上の行列は同じ変換を表す $GF(2)$ 上の行列に一对一で対応させることができる。よって $GF(2)$ 上の行列はより一般的な表現を表しているといえる。 $GF(2)$ 上では行と列の本数が $mn$ 本ずつと、 $GF(2^n)$ の場合と比べて $n$ 倍となる。このため $GF(2^n)$ 上の行列の1行目は、 $GF(2)$ 上の行列の1から $n$ 行目に対応し、1列目は1から $n$ 列目に対応している。つまり $i$ 行目は $(i-1)+1$ 行目から $(i-1)+n$ 行目に対応し、 $i$ 列目は $(i-1)+1$ 列目から $(i-1)+n$ 列目に対応している。よって、 $GF(2^n)$ 上の行や列を取り出してくる操作には、 $GF(2)$ 上で定義される行列を用いる場合は、対応する $n$ 行分もしくは $n$ 列分を取り出すという操作に対応させればよい。 $GF(2^n)$ 上の $m$ 本の行や列を取り出す操作は、 $GF(2)$ 上では $n$ 本の行や列を $m$ 回取り出すことになり、結果として $mn \times mn$ の行列を得ることができる。以上の対応付けにより、 $GF(2)$ 上で定義される行列に容易に拡張することができる。

#### 【0186】

最後に、暗号処理を実行する暗号処理装置としてのICモジュール600の構成例を図18に示す。上述の処理は、例えばPC、ICカード、リーダライタ、その他、様々な情報処理装置において実行可能であり、図18に示すICモジュール600は、これら様々な機器に構成することが可能である。

#### 【0187】

図18に示すCPU(Central processing Unit)601は、暗号処理の開始や、終了、データの送受信の制御、各構成部間のデータ転送制御、その他の各種プログラムを実行するプロセッサである。メモリ602は、CPU601が実行するプログラム、あるいは演算パラメータとしての固定データを格納するROM(Read-Only-Memory)、CPU601の処理において実行されるプログラム、およびプログラム処理において適宜変化するパラメータの格納エリア、ワーク領域として使用されるRAM(Random Access Memory)等からなる。また、メモリ602は暗号処理に必要な鍵データ等の格納領域として使用可能である。データ等の格納領域は、耐タンパ構造を持つメモリとして構成されることが好ましい。

#### 【0188】

暗号処理部603は、例えば上述したFeistel型共通鍵ブロック暗号処理アルゴリズムに従った暗号処理、復号処理等を実行する。なお、ここでは、暗号処理手段を個別モジュールとした例を示したが、このような独立した暗号処理モジュールを設けず、例えば暗号処理プログラムをROMに格納し、CPU601がROM格納プログラムを読み出して実行するように構成してもよい。

#### 【0189】

乱数発生器604は、暗号処理に必要な鍵の生成などにおいて必要となる乱数の発

生処理を実行する。

#### 【0190】

送受信部605は、外部とのデータ通信を実行するデータ通信処理部であり、例えばリーダーライタ等、ICモジュールとのデータ通信を実行し、ICモジュール内で生成した暗号文の出力、あるいは外部のリーダーライタ等の機器からのデータ入力などを実行する。

#### 【0191】

以上、特定の実施例を参照しながら、本発明について詳解してきた。しかしながら、本発明の要旨を逸脱しない範囲で当業者が該実施例の修正や代用を成し得ることは自明である。すなわち、例示という形態で本発明を開示してきたのであり、限定的に解釈されるべきではない。本発明の要旨を判断するためには、特許請求の範囲の欄を参酌すべきである。

#### 【0192】

なお、明細書中において説明した一連の処理はハードウェア、またはソフトウェア、あるいは両者の複合構成によって実行することが可能である。ソフトウェアによる処理を実行する場合は、処理シーケンスを記録したプログラムを、専用のハードウェアに組み込まれたコンピュータ内のメモリにインストールして実行させるか、あるいは、各種処理が実行可能な汎用コンピュータにプログラムをインストールして実行させることが可能である。

#### 【0193】

例えば、プログラムは記録媒体としてのハードディスクやROM (Read Only Memory) に予め記録しておくことができる。あるいは、プログラムはフレキシブルディスク、CD-ROM (Compact Disc Read Only Memory)、MO (Magneto optical) ディスク、DVD (Digital Versatile Disc)、磁気ディスク、半導体メモリなどのリムーバブル記録媒体に、一時的あるいは永続的に格納（記録）しておくことができる。このようなリムーバブル記録媒体は、いわゆるパッケージソフトウェアとして提供することができる。

#### 【0194】

なお、プログラムは、上述したようなリムーバブル記録媒体からコンピュータにインストールする他、ダウンロードサイトから、コンピュータに無線転送したり、LAN (Local Area Network)、インターネットといったネットワークを介して、コンピュータに有線で転送し、コンピュータでは、そのようにして転送されてくるプログラムを受信し、内蔵するハードディスク等の記録媒体にインストールすることができる。

#### 【0195】

なお、明細書に記載された各種の処理は、記載に従って時系列に実行されるのみならず、処理を実行する装置の処理能力あるいは必要に応じて並列的にあるいは個別に実行されてもよい。また、本明細書においてシステムとは、複数の装置の論理的集合構成であり、各構成の装置が同一筐体内にあるものには限らない。

#### 【産業上の利用可能性】

#### 【0196】

上述したように、本発明の構成によれば、非線形変換部および線形変換部を有するSPN型のF関数を、複数ラウンド繰り返し実行するFeistel型共通鍵ブロック暗号処理において、複数ラウンド各々に対応するF関数の線形変換処理を、正方MDS (Maximum Distance Separable) 行列を適用した線形変換処理として実行するとともに、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列： $L_a$ 、 $L_b$ を適用し、かつ該正方MDS行列の逆行列： $L_a^{-1}$ 、 $L_b^{-1}$ を構成する列ベクトルから任意に選択したm個の列ベクトルによって構成する行列が線形独立であるか、あるいは正方MDS行列を構成する性質とした正方MDS行列による線形変換処理を実行する構成としたので、共通鍵ブロック暗号における線形攻撃に対する耐性が向上し、暗号鍵等の解析困難性が高まることとなり、安全性の高い暗号処理が実現される。従って、鍵解析困難性を高め、安全性の要求される暗号処理実行装置において適用可能である。

## 【0197】

さらに、本発明の構成によれば、非線形変換部および線形変換部を有するSPN型のF関数を、複数ラウンド繰り返し実行するFeistel型共通鍵ブロック暗号処理において、複数ラウンド各々に対応するF関数の線形変換処理を、正方MDS（Maximum Distance Separable）行列を適用した線形変換処理として実行するとともに、少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において異なる正方MDS行列を適用する構成とし、これらの正方MDS行列自身が、線形独立性を示すか、または正方MDS行列を構成する構成としたので、アクティブSボックスの寄与による同時差分キャンセルの発生しないことが保証され、共通鍵ブロック暗号における差分攻撃に対する強度指標のひとつである暗号化関数全体でのアクティブSボックスの最少数を大きくすることが可能となる。本構成により、線形攻撃、差分攻撃の双方に対して耐性が向上し、より安全性の高い暗号処理が実現される。従って、鍵解析困難性を高め、安全性の要求される暗号処理実行装置において適用可能である。

## 【図面の簡単な説明】

## 【0198】

【図1】 Feistel構造を持つ代表的な共通鍵ブロック暗号の構成を示す図である。

【図2】 ラウンド関数部として設定されるF関数の構成について説明する図である。

【図3】 線形変換部において、線形変換処理に適用する正方行列の例を示す図である。

【図4】  $m=8$ 、 $n=8$ の128bitブロック暗号における3段の同時差分キャンセルの様子を説明する図である。

【図5】 F関数の線形変換部において、正方行列による線形変換が実行されて、F関数出力差分 $\Delta Y_i$ を生成する具体例を説明する図である。

【図6】  $m=8$ 、 $n=8$ の128bitブロック暗号における5段の同時差分キャンセルの様子を説明する図である。

【図7】 共通鍵ブロック暗号における任意段の同時差分キャンセルの定義を説明する図である。

【図8】 正方MDS行列の一例を示す図である。

【図9】 本発明に係る共通鍵ブロック暗号処理アルゴリズムにおける各ラウンドのF関数の線形変換行列としての正方MDS行列設定例を説明する図である。

【図10】 本発明に係る共通鍵ブロック暗号処理アルゴリズムにおける各ラウンドのF関数の線形変換行列としての正方MDS行列設定処理シーケンスを説明するフロー図である。

【図11】 各ラウンドのF関数に設定する線形変換行列である正方MDS行列の生成手法として、差分攻撃に対する耐性向上を実現する正方MDS行列生成処理例a1を説明するフロー図である。

【図12】 各ラウンドのF関数に設定する線形変換行列である正方MDS行列の生成手法として、差分攻撃に対する耐性向上を実現する正方MDS行列生成処理例a2を説明するフロー図である。

【図13】 各ラウンドのF関数に設定する線形変換行列である正方MDS行列の生成手法として、差分攻撃に対する耐性向上を実現する正方MDS行列生成処理例a3を説明するフロー図である。

【図14】 各ラウンドのF関数に設定する線形変換行列である正方MDS行列の生成処理例a3の具体的手法を説明する図である。

【図15】 各ラウンドのF関数に設定する線形変換行列である正方MDS行列の生成手法として、線形攻撃に対する耐性向上を実現する正方MDS行列生成処理例b1を説明するフロー図である。

【図16】 各ラウンドのF関数に設定する線形変換行列である正方MDS行列の生成手法として、線形攻撃に対する耐性向上を実現する正方MDS行列生成処理例b2を



説明するフロー図である。

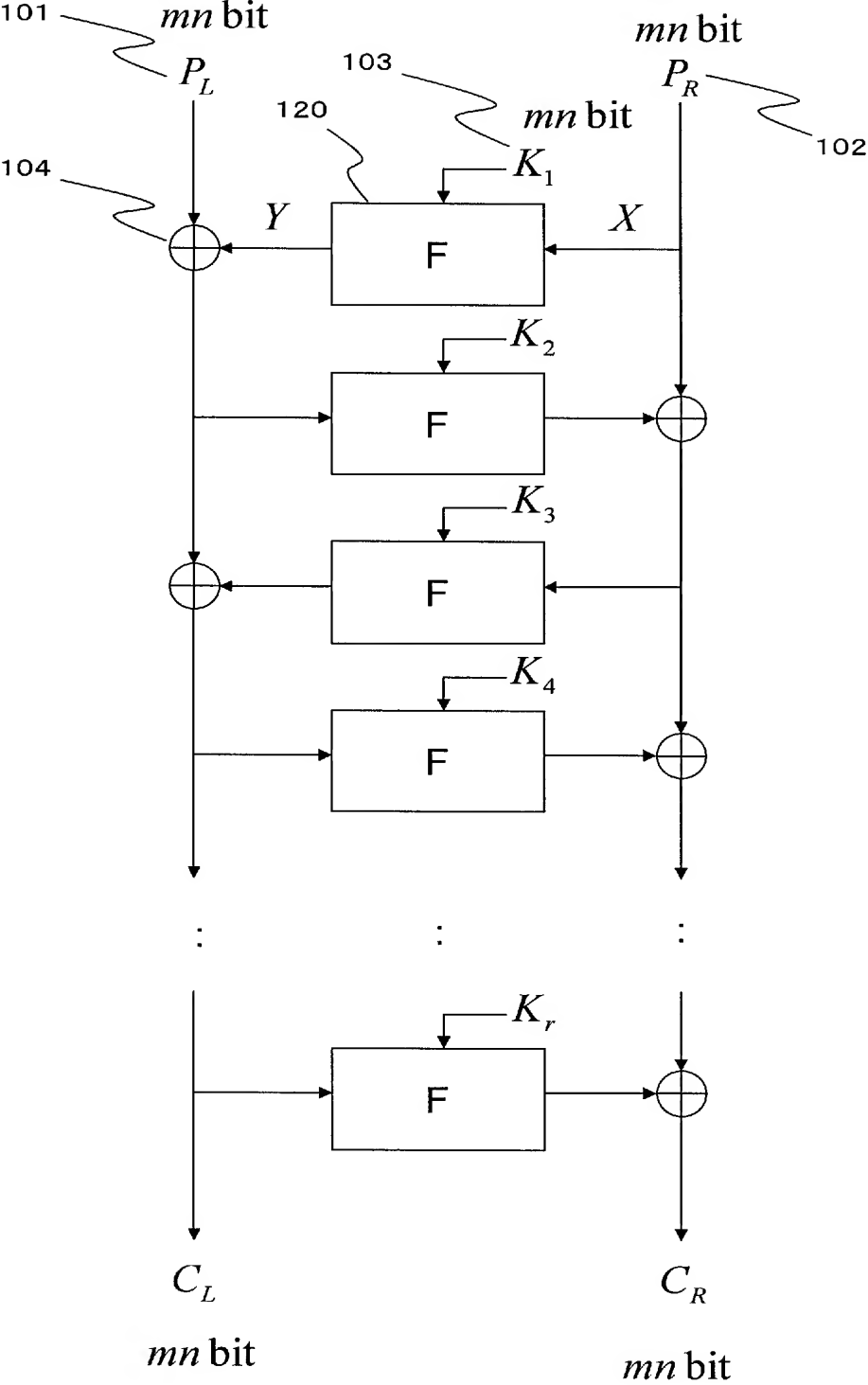
【図 1 7】 各ラウンドの F 関数に設定する線形変換行列である正方 M D S 行列の生成手法として、差分攻撃および線形攻撃に対する耐性向上を実現する正方 M D S 行列生成処理例を説明するフロー図である。

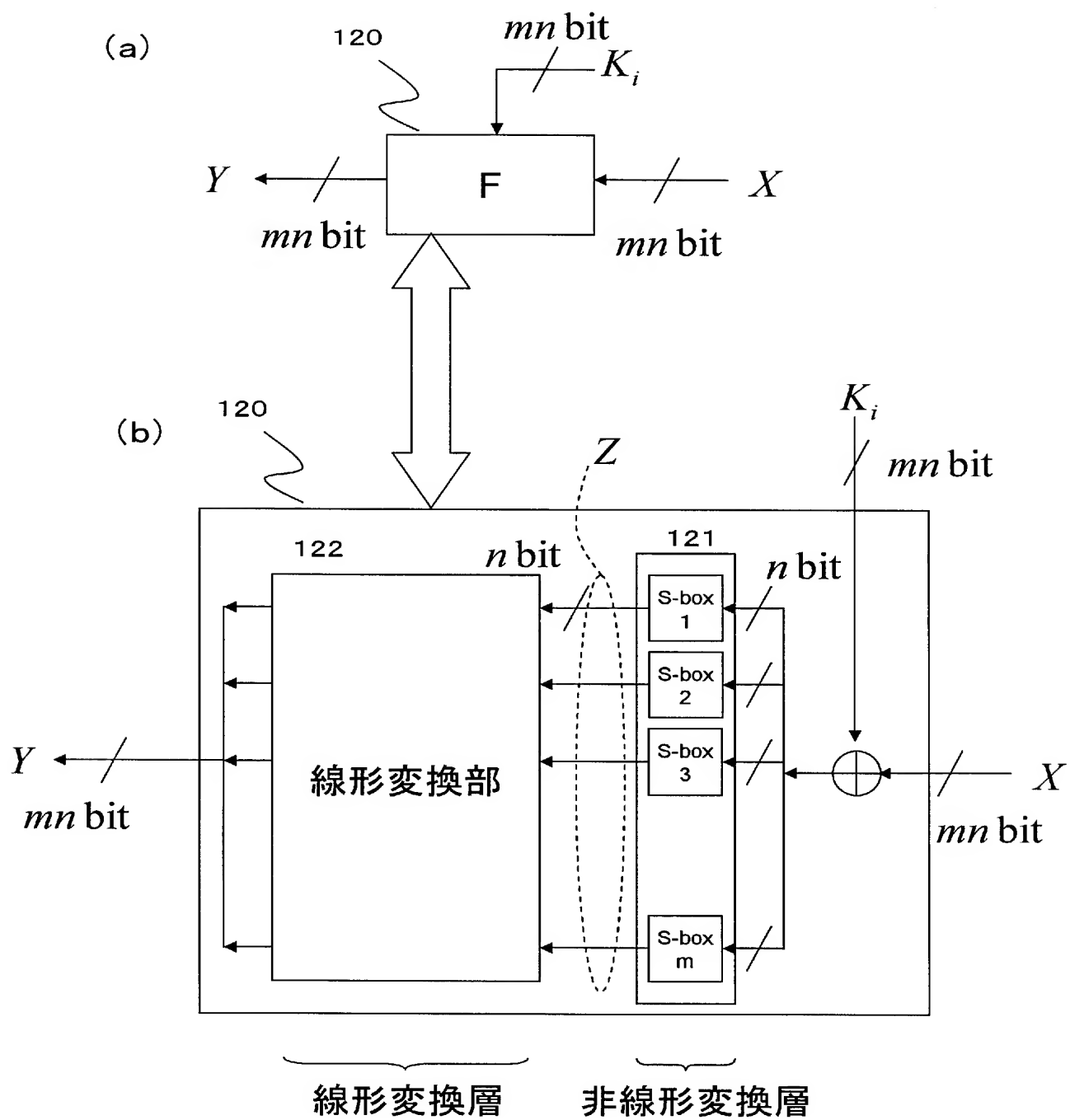
【図 1 8】 本発明にかかる暗号処理を実行する暗号処理装置としての I C モジュールの構成例を示す図である。

【符号の説明】

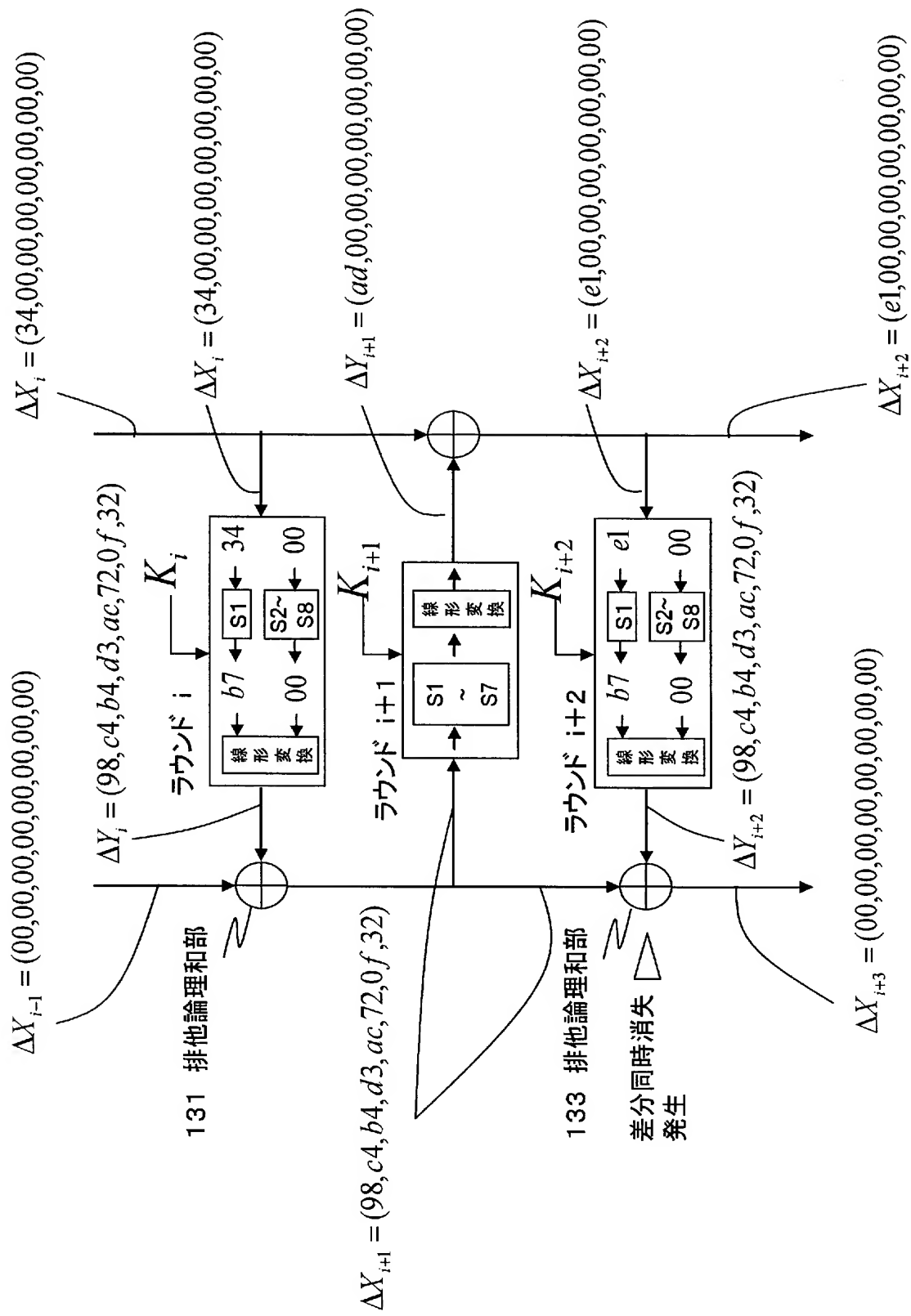
【 0 1 9 9 】

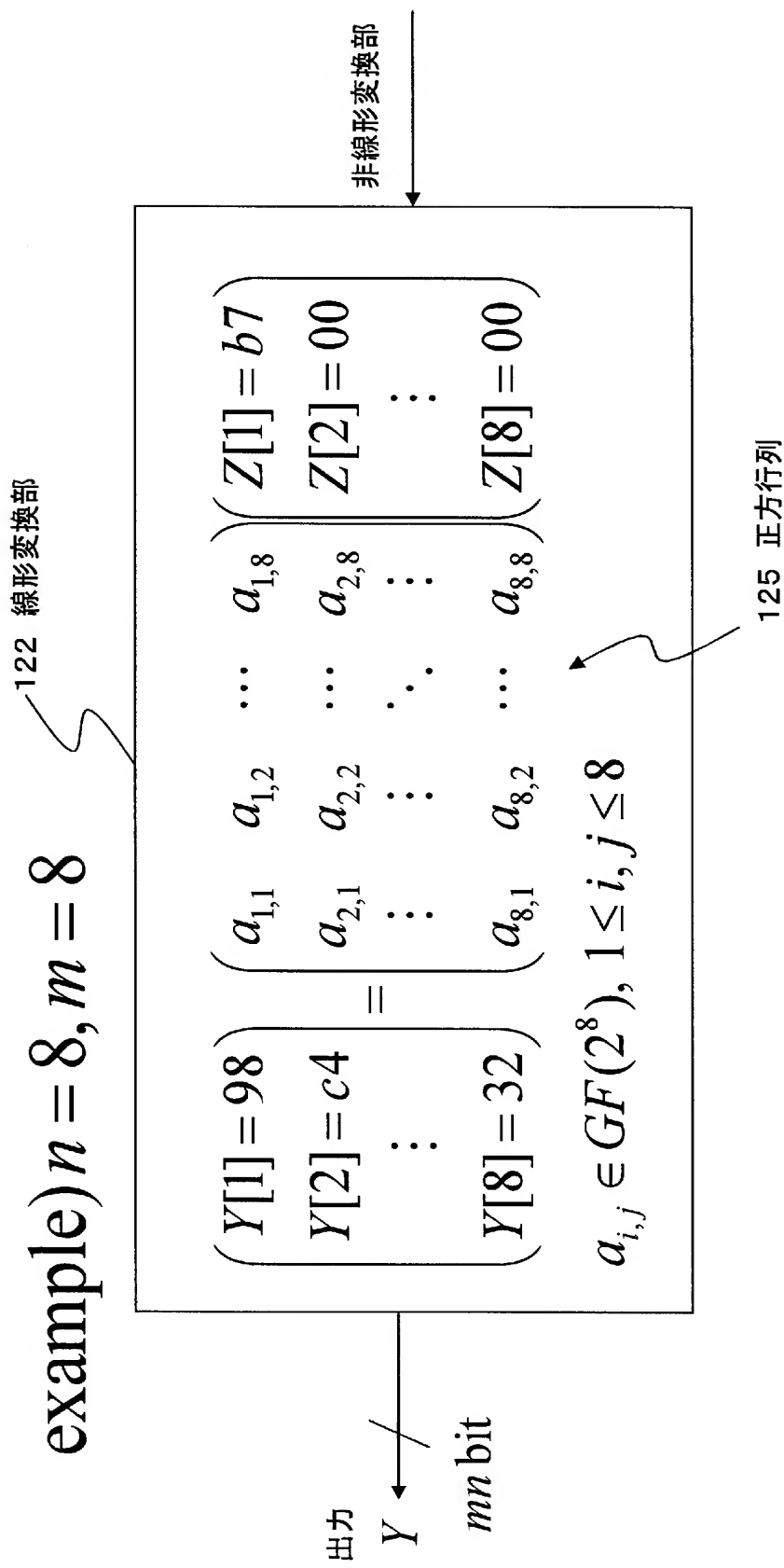
- 1 0 1 入力データ  $P_L$  (Plain-Left)
- 1 0 2 入力データ  $P_R$  (Plain-Right)
- 1 0 3 ラウンド鍵
- 1 0 4 排他的論理和部
- 1 2 0 F 関数
- 1 2 1 S ボックス (S - b o x )
- 1 2 2 線形変換部
- 1 2 5 正方行列
- 1 3 1 , 1 3 3 排他的論理和部
- 1 4 1 ~ 1 4 5 排他的論理和部
- 4 0 1 F 関数
- 4 0 2 正方 M D S 行列
- 6 0 0 I C モジュール
- 6 0 1 C P U (Central processing Unit)
- 6 0 2 メモリ
- 6 0 3 暗号処理部
- 6 0 4 乱数発生器
- 6 0 5 送受信部

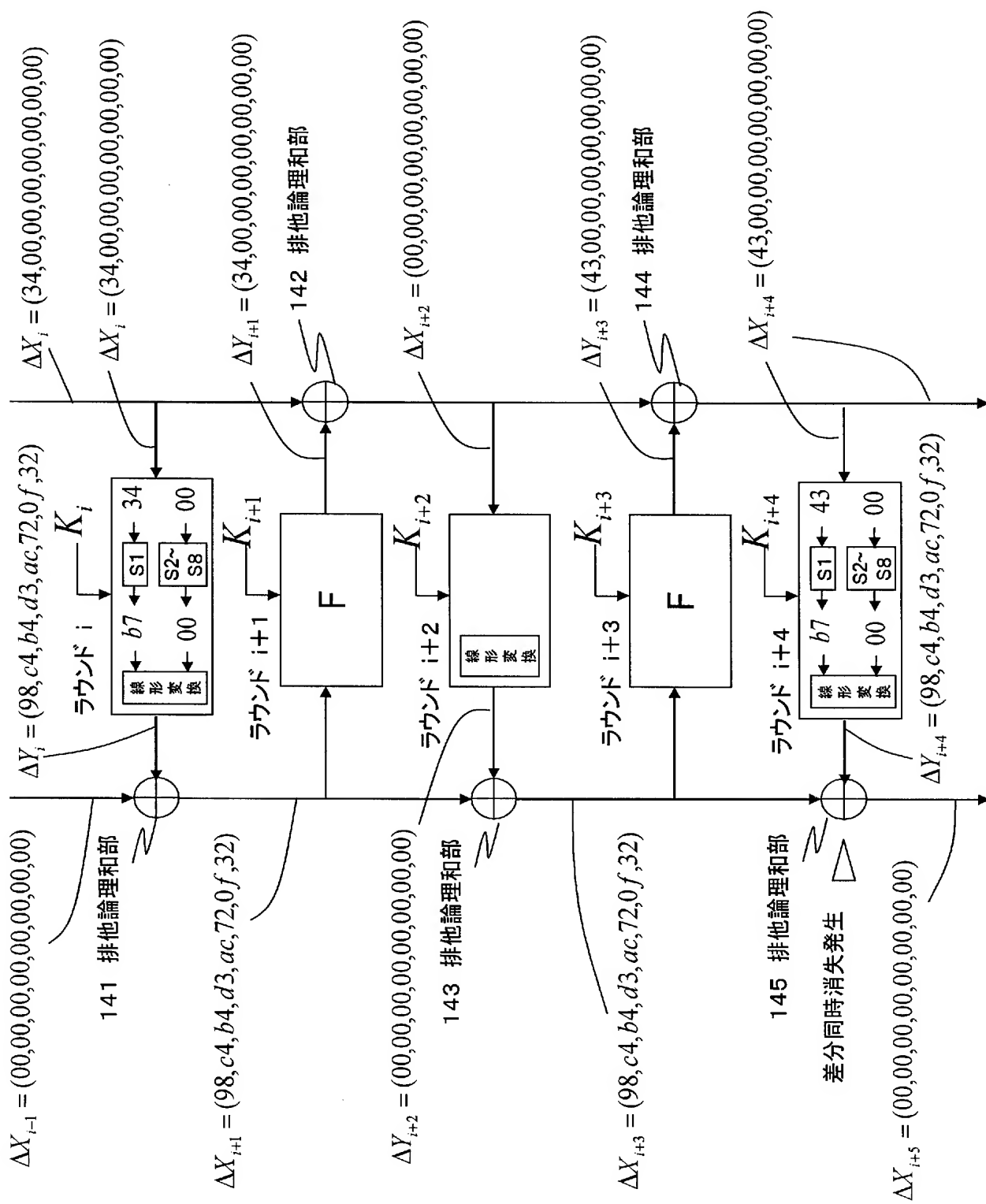




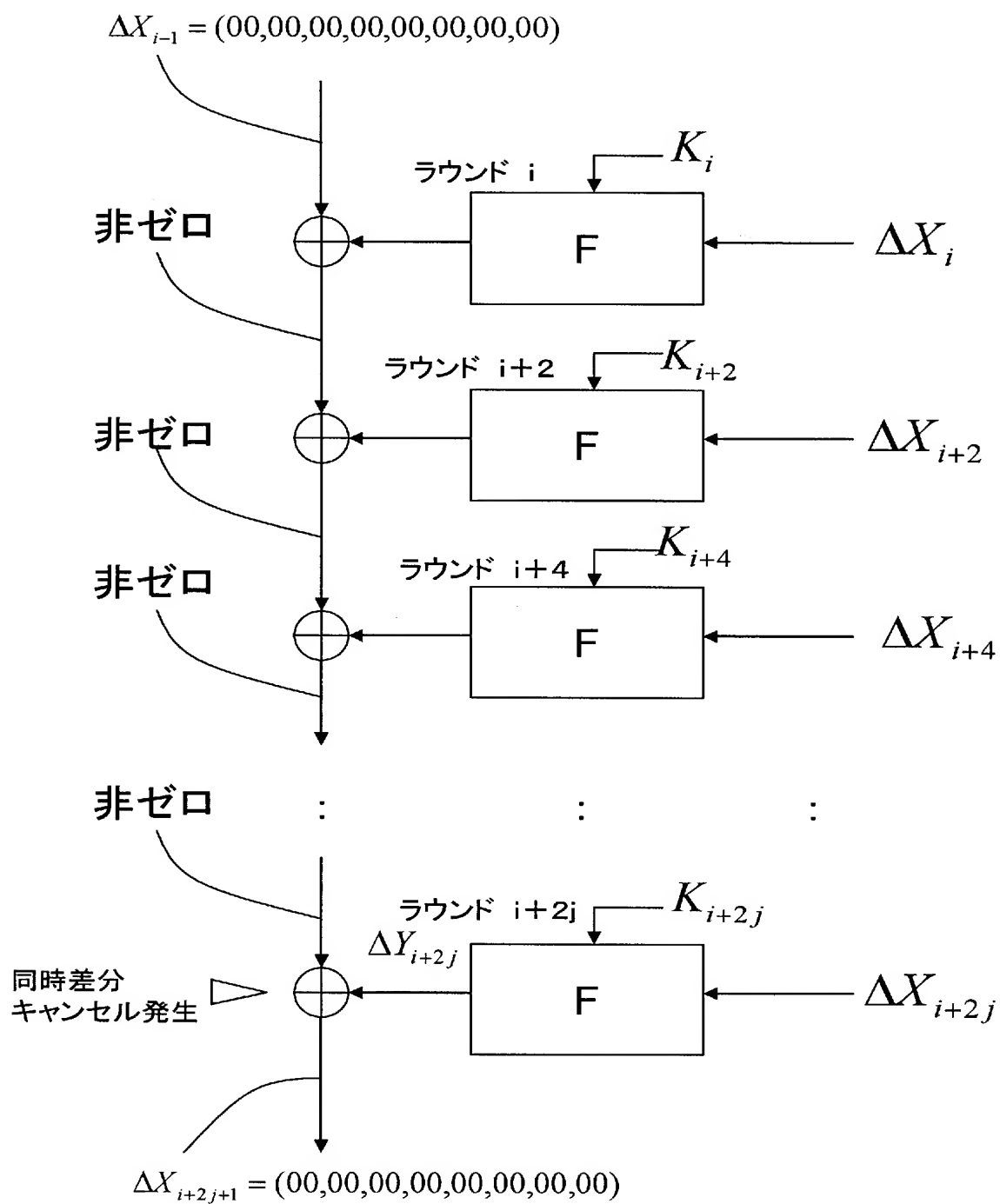








【図 7】



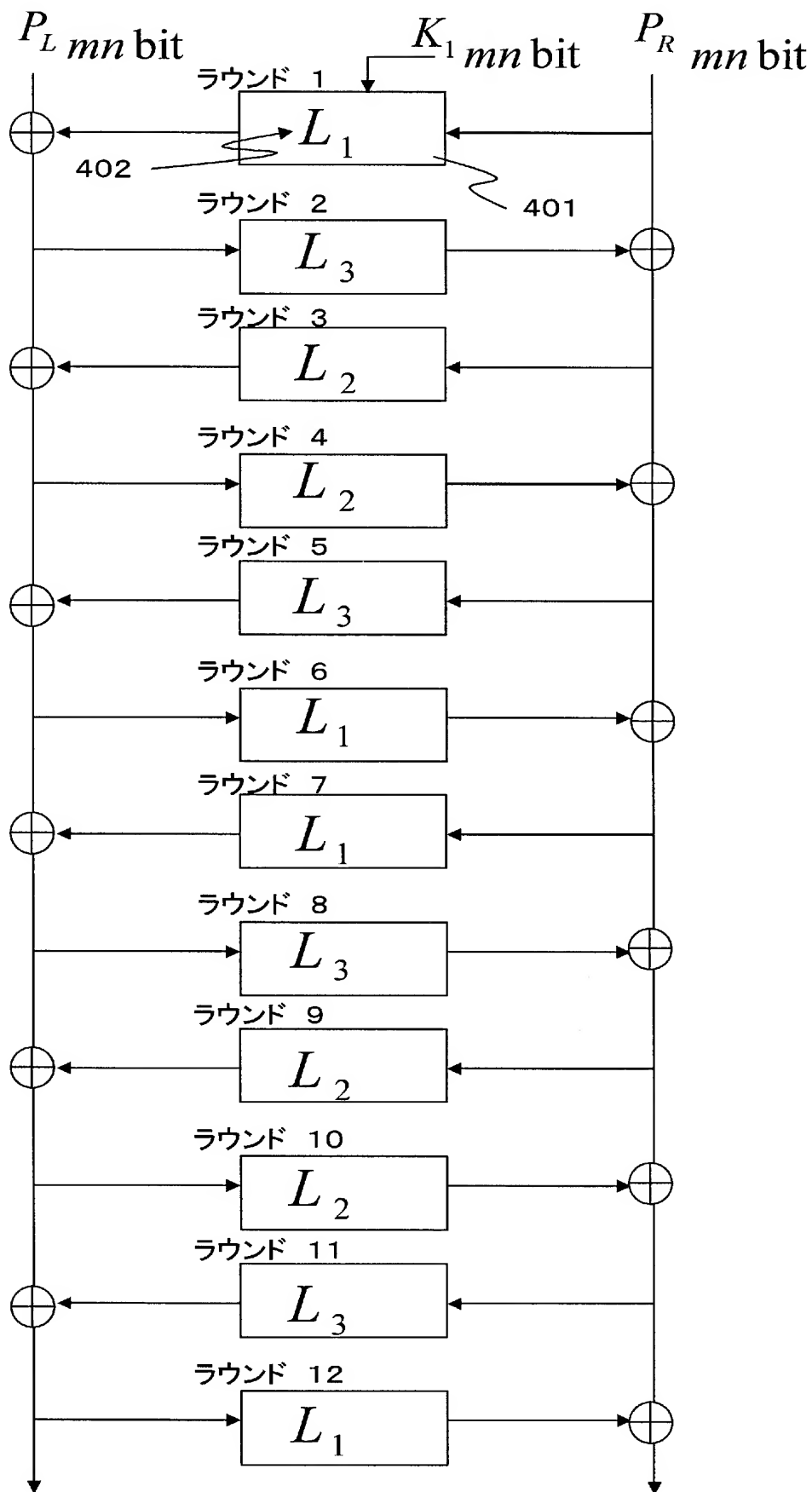


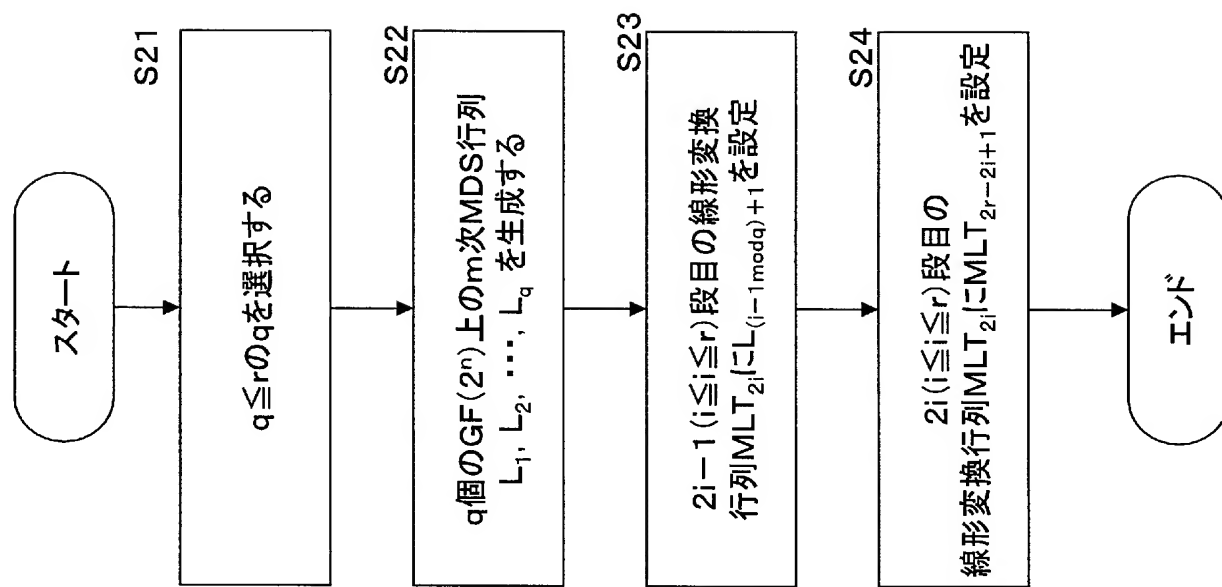
example)  $n = 8, m = 8$

$9d$	$b4$	$d3$	$5d$	$84$	$ae$	$ec$	$b9$
$29$	$34$	$39$	$60$	$5c$	$81$	$25$	$13$
$67$	$6a$	$d2$	$e3$	$4b$	$db$	$9d$	$4$
$8e$	$d7$	$e6$	$1b$	$8b$	$9e$	$3a$	$91$
$d9$	$e5$	$4d$	$dd$	$c6$	$5$	$f0$	$ad$
$2a$	$f7$	$67$	$72$	$b1$	$7$	$f2$	$27$
$42$	$e6$	$a0$	$4$	$f1$	$4$	$7d$	$8c$
$55$	$63$	$fa$	$51$	$c$	$d9$	$28$	$d6$

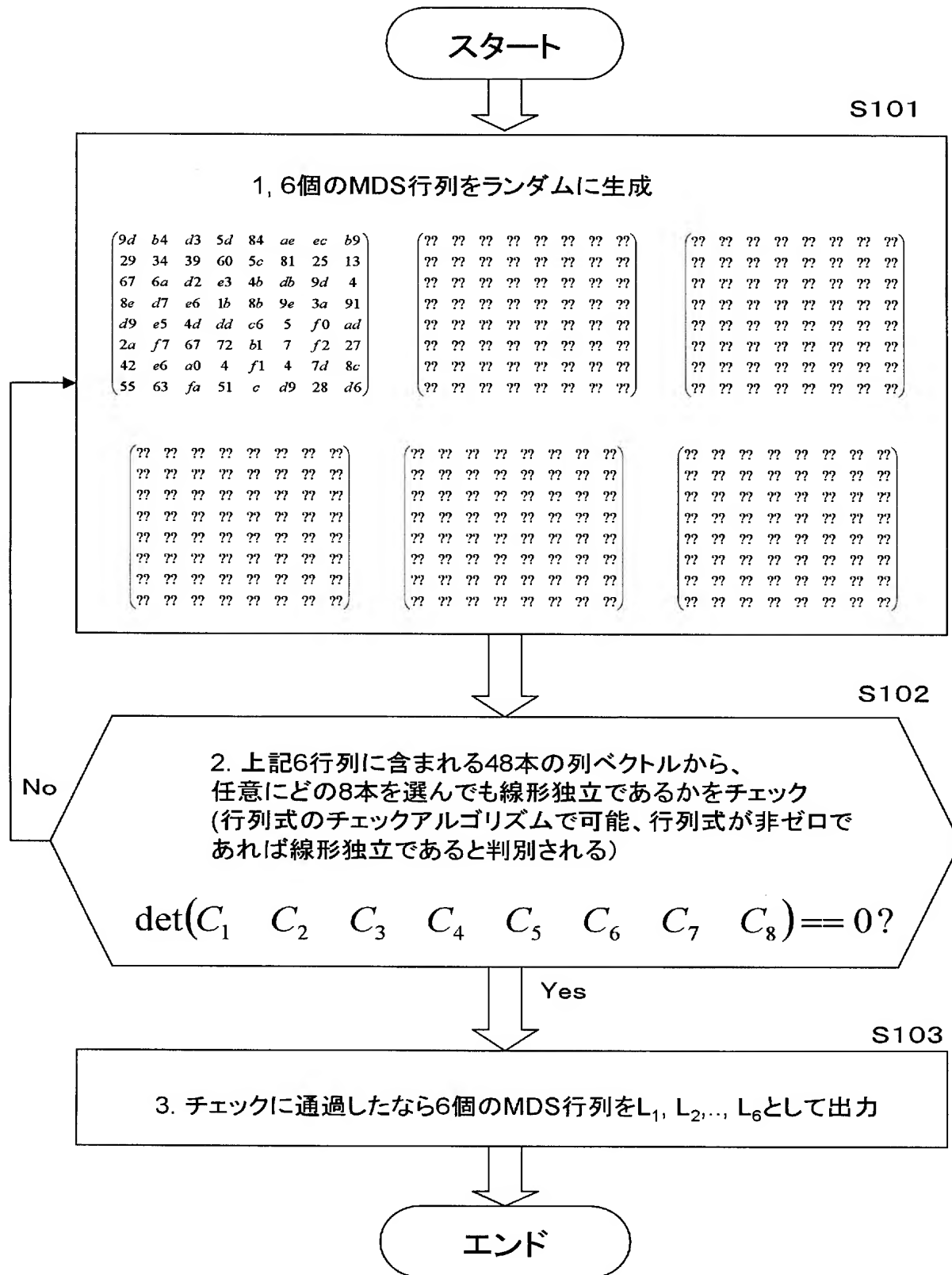
【図 9】

$r = 6, q = 3$   
の設定例

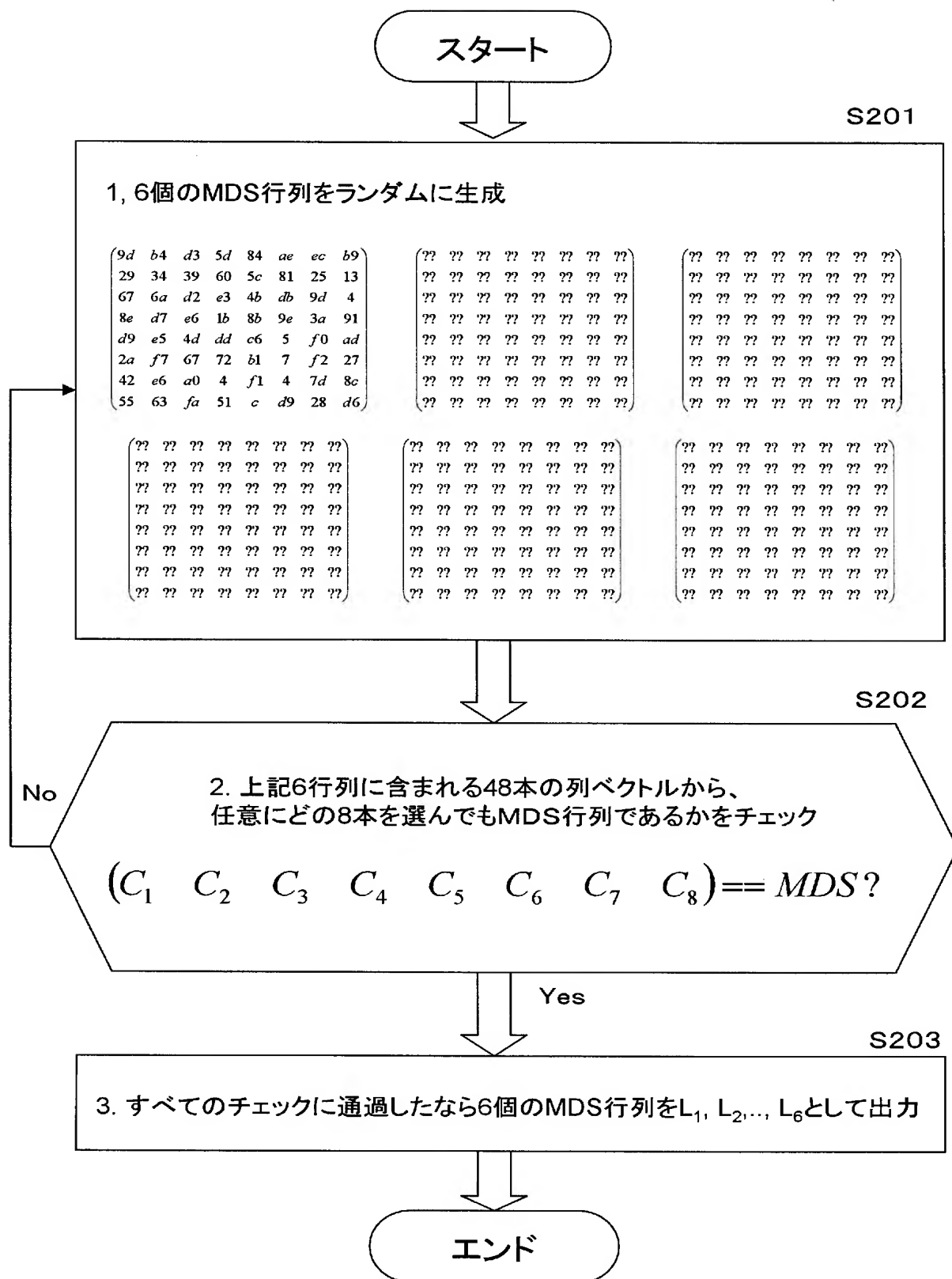




$q = 6, n = 8, m = 8$  の場合



$q = 6, n = 8, m = 8$  の場合

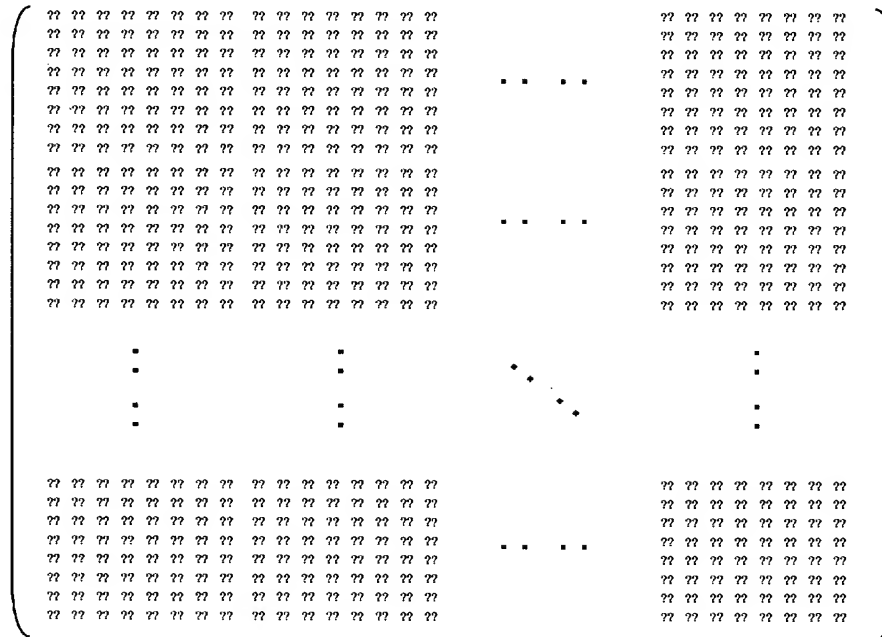


$q = 6, n = 8, m = 8$   
の場合

スタート

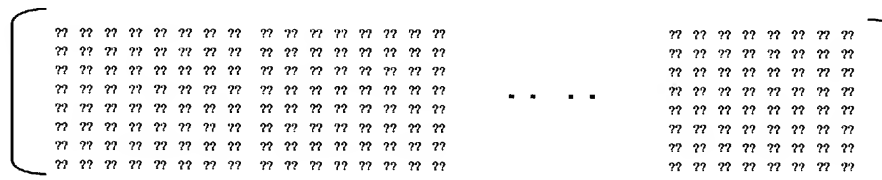
S301

48x48のMDS行列Mを生成



S302

上記行列から任意の8本の行ベクトルを選択しM'とする

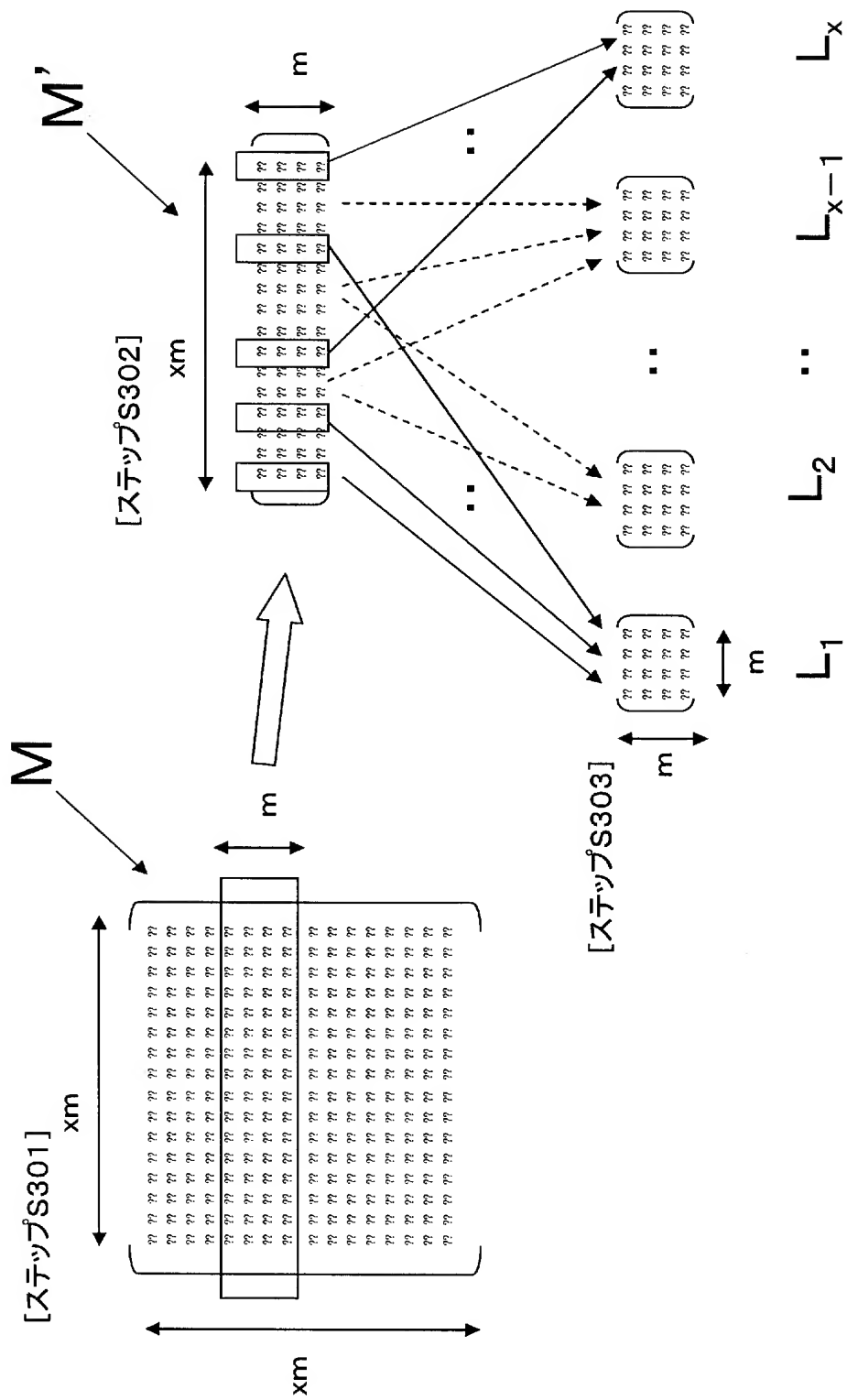


S303

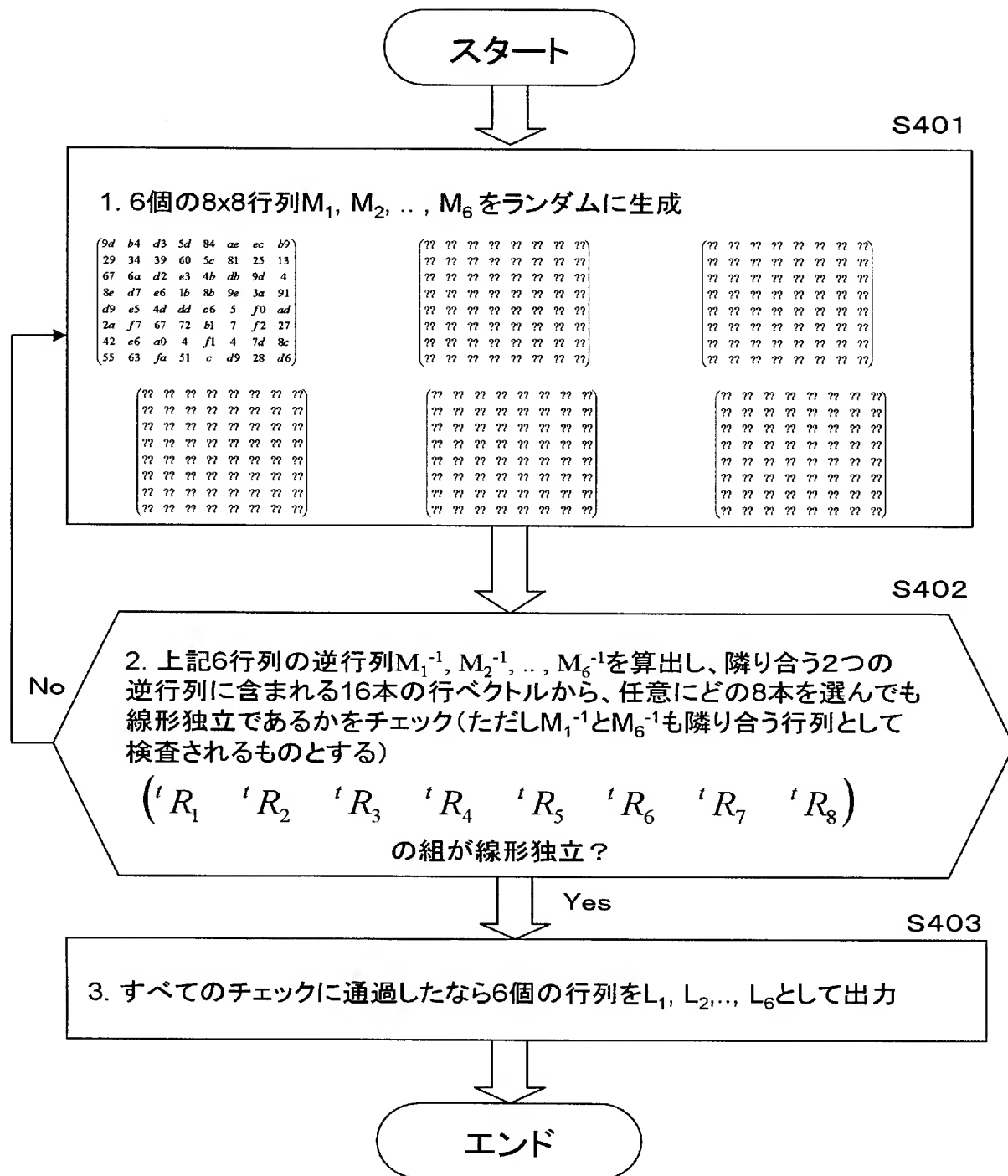
M'の48本の列ベクトルを8本ずつ6つのグループに分割し、8x8の行列を作成  
それらをL1, L2,..., L6として出力



エンド

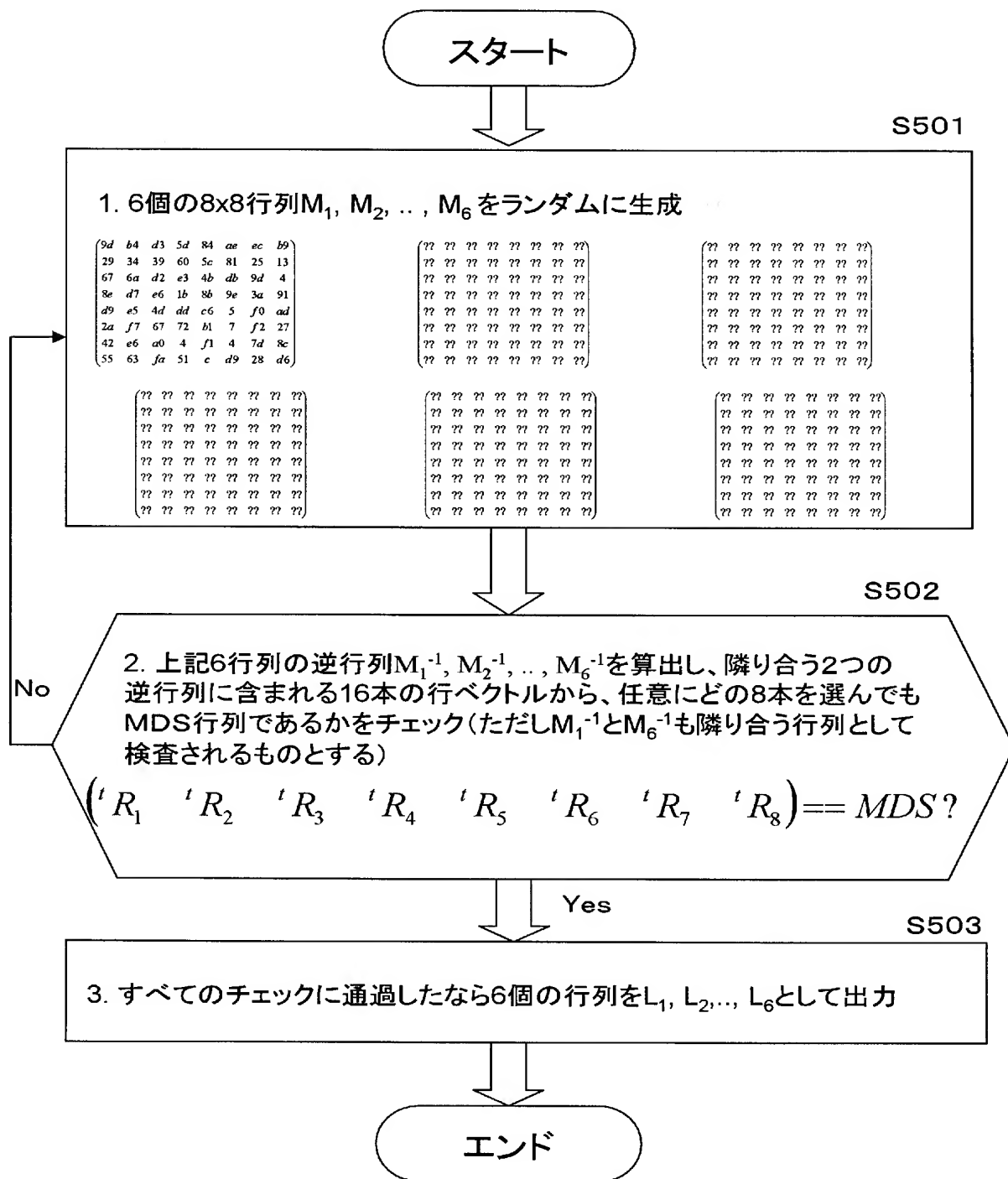


$q = 6, n = 8, m = 8$  の場合





$q = 6, n = 8, m = 8$  の場合



$q = 6, n = 8, m = 8$   
の場合

スタート

S601

1. 6個のMDS行列  $M_1, M_2, \dots, M_6$  をランダムに生成

9d	b4	d3	5d	84	ae	ec	b9
29	34	39	60	5c	81	25	13
67	6a	d2	e3	4b	db	9d	4
8e	d7	e6	1b	8b	9e	3a	91
d9	e5	4d	dd	c6	5	f0	ad
2a	f7	67	72	b1	7	f2	27
42	e6	a0	4	f1	4	7d	8c
55	63	fa	51	c	d9	28	d6

??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??

??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??

??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??

??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??

??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??
??	??	??	??	??	??	??	??

S602

No

2. 上記6行列に含まれる48本の列ベクトルから、  
任意にどの8本を選んでもMDS行列であるかをチェック

$(C_1 \ C_2 \ C_3 \ C_4 \ C_5 \ C_6 \ C_7 \ C_8) == MDS?$

Yes

S603

No

3. 上記6行列の逆行行列を算出し、隣り合う2つの逆行行列に含まれる  
16本の行ベクトルから、任意にどの8本を選んでもMDS行列であるかを  
チェック(ただし  $M_1^{-1}$  と  $M_6^{-1}$  も隣り合う行列として検査されるものとする)

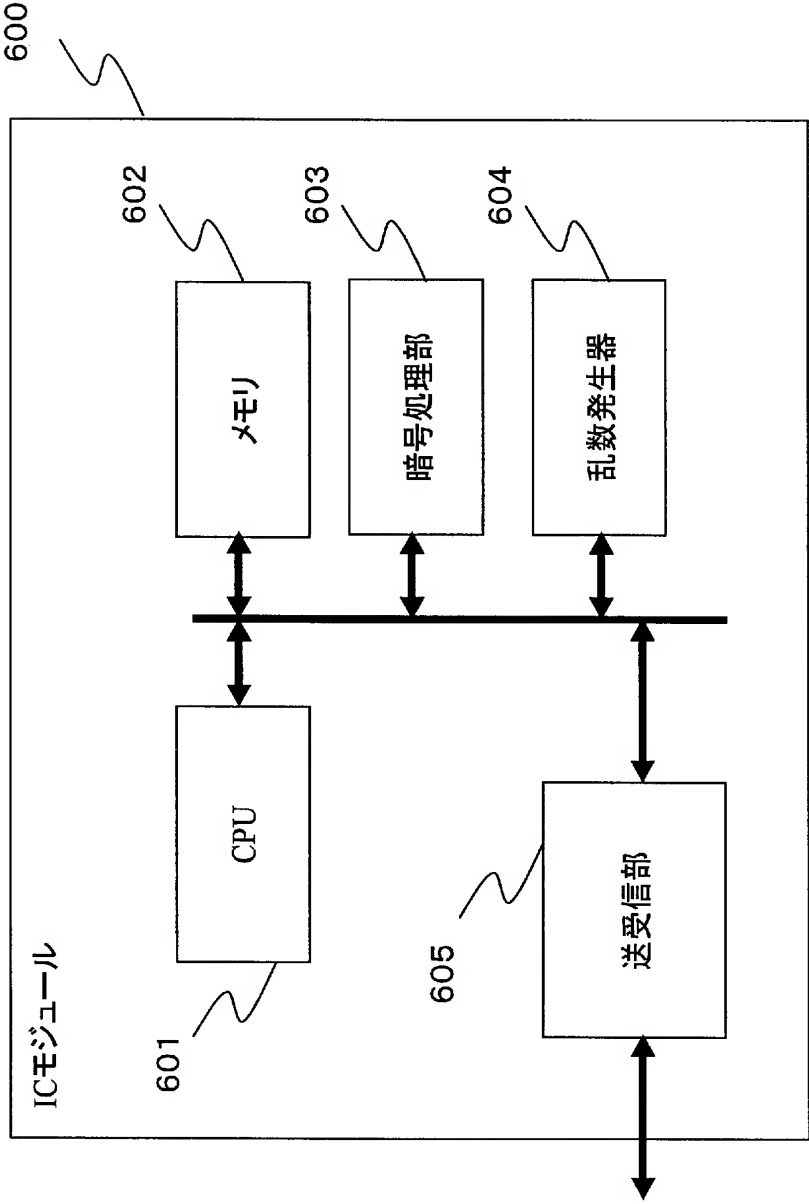
$({}^tR_1 \ {}^tR_2 \ {}^tR_3 \ {}^tR_4 \ {}^tR_5 \ {}^tR_6 \ {}^tR_7 \ {}^tR_8) == MDS?$

Yes

S604

4. すべてのチェックに通過したなら6個のMDS行列を  $L_1, L_2, \dots, L_6$  として出力

エンド



【書類名】 要約書

【要約】

【課題】 解析困難性を高めた、安全性の高い暗号処理装置および方法を実現する。

【解決手段】 非線形変換部および線形変換部を有するSPN型のF関数を、複数ラウンド繰り返し実行するFeistel型共通鍵ブロック暗号処理において、複数ラウンド各々に対応するF関数の線形変換処理を、正方MDS(Maximum Distance Separable)行列を適用した線形変換処理とする。少なくとも連続する偶数ラウンドおよび連続する奇数ラウンドの各々において設定される正方MDS行列の逆行列に含まれる任意のm個の列ベクトルが正方MDS行列である設定とする。本構成により、共通鍵ブロック暗号における線形攻撃に対する耐性の向上した暗号処理が実現される。

【選択図】 図17

## 出願人履歴

0 0 0 0 0 2 1 8 5

19900830

新規登録

5 9 7 0 6 2 9 9 3

東京都品川区北品川 6 丁目 7 番 3 5 号

ソニー株式会社